Печат: ДИТА ММ ООД
гр. София 1309, ж.к. Захарна фабрика, ул. Кукуш 1
тел.: 821 12 19, 821 12 39
e-mail: office@dita-mm.com

Contents

Preface

Third INTERNATIONAL CONFERENCE on
INFORMATION SYSTEMS & GRID TECHNOLOGIES
28 - 29 May 2009, Sofia, Bulgaria
organized by University of Sofia “St. Kliment Ohridski”
and BulAIS - Bulgarian Chapter of AIS
Dedicated to 120 years of Faculty of Mathematics and Informatics at University of Sofia

CONFERENCE TRACKS & TOPICS

Information Systems
- Storage Management
- Data Management and Transport
- Data Warehouse
- Data Mining/Streams
- Caching and Query Optimization
- Metadata & Sampling
- Mobile & Wireless Access to Information Systems

Intelligent Systems
- Knowledge & Data Management
- Knowledge Engineering
- Information Retrieval
- Voice and Image Processing
- Temporal Reasoning
- Web-based Technology and AI
- Digital Libraries

Grid Technologies
- Grid Information, Resource and Workflow Monitoring Services
- Resource Management, Scheduling and Load-balancing
- Systems, Tools and Environments
- Communication Paradigms and Technologies
- Service-centric Architectures and Cloud
- Multicast and Web Services
- Security and User Management Multimedia, Teleimmersion and Collaborative Applications

ORGANIZATION COMMITTEE
- assoc. prof. Vladimir DIMITROV
- assoc. prof. Maria NISHEVA
- assoc. prof. Kalinka KALOYANOVA
- assoc. prof. Vasil GEORGIEV

CONTACT
- http://isgt.fmi.uni-sofia.bg
- e-mail: mailto:isgt@fmi.uni-sofia.bg

This conference has been sponsored by Scientific Research Fund of University of Sofia under contract 200/2009, IBM Bulgaria and TechnoLogica. Thanks to our sponsors!

In the scope of the conference has been held a Workshop on DB2 for students. Materials of the workshop are not published.

These materials are prepared and edited by assoc. prof. Vladimir Dimirtrov.

Image and Video Processing Methods for High Performance Data Management Systems

Lev N. Korolev[1], Nina N. Popova[1], Oxana V. Dzhosan[1]

[1] Moscow State University, dep. of Computational Mathematics and Cybernetics,
Leninskie Gory, MSU, 2-nd educational building, VMK Faculty,
119992, Moscow, Russia
{krlv, popova}@ cs.msu.su, oxanad@mail.ru

Abstract. In this article parallel image and video processing methods for large scale scientific data visualization system are described. Proposed system uses hierarchical distribution of data processing modules as the parts of computing system according to modules computational complexity and effectiveness of processing. Paper describes methods for image and video enhancement, video compression and video up-sampling. Enhancement method is based on jagged edges processing. Video compression method is based on block oriented video coding with color pallet. Adaptive up-sapling method uses wavelet and triangulation for interpolation. IBM Blue Gene /P is used as main test system.

Keywords: Image and video processing, scientific data visualization, high performance computing, Blue Gene /P, parallel video processing

1 Introduction

One of the main problems in high performance computing is how to analyze large scale data that is received during the computation. Reduction of data size that is transmitted from the computational complex to local PC workstation for analysis is important problem. One of the ways for such reduction is large scale data visualization and analysis of image or video.

On of the tasks in large scale data visualization is how to reduce the time of frame rendering but in the same time not to damage visual quality of the image. So this paper proposes methods for image and video processing for data size reduction. One of the solutions is to render image of small size and then to apply interpolation method to up-size frame to the required ratio. One of the suggested methods concerns to the way of image and video data processing and, in particular, to the way of image and video sequence up-sampling (up-sizing). One more method is video compression method that is based on block truncation coding and is used for video encoding before transmitting it to local work station for analysis. Also in this paper it is suggested a method of video enhancement for video postprocessing based on jagged edges reduction. These methods are the modules of large scale scientific data visualization system that is developed in Moscow State University [1].

Different scientific data visualization systems already exist. They provide wide range of visualization possibilities, such as volume and graphic rendering, interactive visualization. The most frequently used general visualization systems, such as Visualization Toolkit (VTK) [2], ParaView system [3], VisIt [4] and so on, are rather effective and in most cases of processing data on PC can produce good results. But when using such systems on high performance complexes they loose their effectiveness and strong modification and adaptation are required. This conclusion can be done after experience with installation of VTK, ParaView and VisIt systems on IB Blue Gene /P complex.

The task of visualization of large scale data produces on Blue Gene /P was studied in [5]. There were proposed four different strategies for parallel visualization on Blue Gene /P complex and their effectiveness was analyzed. In Argonne National Laboratory [6] Blue Gene /P system was add with installation of NVIDIA Quadro Plex S4 external graphics processing units for fast data visualization.

This paper proposes modules for MSU_VIZ visualization system for Blue Gene /P system solution. MSU_VIZ system is based on hierarchical distribution of video processing method that is based on methods complexity. Proposed system includes parallel method of video rendering in different formats that are used for displaying on different display systems such as 3D displays and multi display complexes. IBM Blue Gene/P system where visualization system tests were performed includes 2048 computational nodes, each node has four PowerPC 450 processors with 850Mhz, system includes 2GB DRAM per each node and 16 I/O nodes. IBM pSeries 55A is connected as front-end.

2 Adaptive image and video interpolation method

A lot of methods of image interpolation are already developed. There is a wide variety of interpolating kernels, such as sinc, Lanzcos, linear, B-spline, Gauss, Hermite, etc. Comprehensive description of generalized interpolation and choice of convolution kernel functions is presented in [7]and [8].

Proposed method uses the combination of wavelets and edge preserving triangulation interpolation method for image and video up-sampling task. Algorithm consists of three main steps: directions and frequency type estimation, interpolation, edge post-processing. Main blocks of pixel type estimation method: derivatives calculation, direction selection, wavelet value calculation and pixel type selection.

First, prevailing edge direction is determined. For this purpose absolute values of difference between neighboring pixels are used. For determination of diagonals absolute values of differences between pixels are utilized, which are located in 5x5 neighborhood of the pixel being interpolated. Let four closest neighbors of interpolated value Y_{kl} are $I_T(i,j)$, $I_T\ (i+1,j)$, $I_T\ (i,j+1)$ and $I_T\ (i+1,j+1)$, where T is a channel, $T \in \{Y, Cb, Cr\}$.

$$D_X = \frac{1}{12} \sum_{\substack{k=-1..1 \\ l=-1..2}} \left| I_Y(i+k+1, j+l) - I_Y(i+k, j+l) \right| \qquad (1)$$

$$D_Y = \frac{1}{12} \sum_{\substack{k=-1..2 \\ l=-1..1}} \left| I_Y(i+k, j+l+1) - I_Y(i+k, j+l) \right| \qquad (2)$$

$$D_{d1} = \frac{1}{9} \sum_{\substack{k=-1..1 \\ l=-1..1}} \left| I_Y(i+k+1, j+l+1) - I_Y(i+k, j+l) \right| \qquad (3)$$

$$D_{d2} = \frac{1}{9} \sum_{\substack{k=-1..1 \\ l=-1..1}} \left| I_Y(i+k+1, j+l) - I_Y(i+k, j+l+1) \right| \qquad (4)$$

After four derivatives are calculated a direction or subdirection is selected which has the maximal value of derivative. So the direction value can take eight different values: {0,1,2,3,4,5,6,7}. Process details are shown on the Fig. 1.

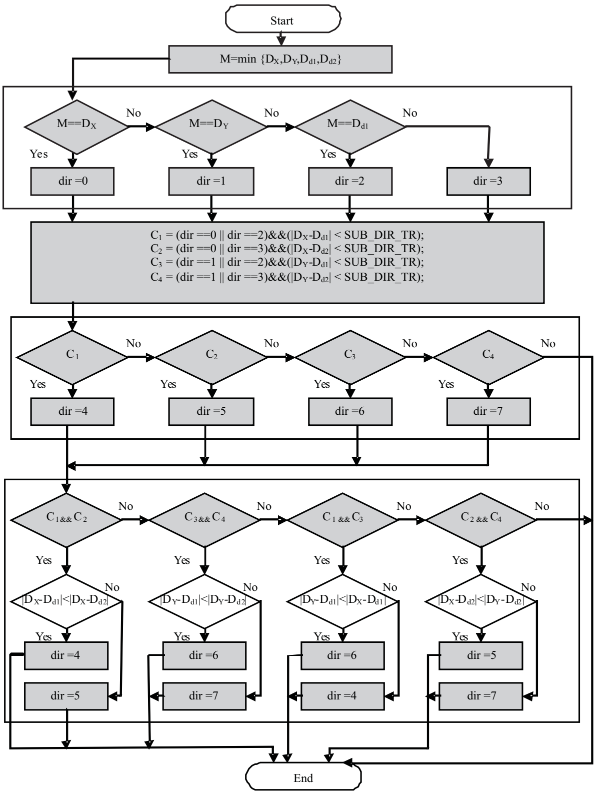


Fig. 1. Scheme of sub directions calculation.

Then frequency area type is estimated using wavelets. Wavelet 5-3 is used in horizontal and vertical direction to calculate area frequency number. Wavelet value W_{HH} is calculated using the following equation:

$$w(T,k,l) = I_T(i+k,\, j+l) - \frac{1}{2}\sum_{h\in\{-1,1\}} I_T(i+k,\, j+l+h) + \frac{1}{2}\sum_{h\in\{-1,1\}} I_T(i+k+h,\, j+l) +$$

$$\frac{1}{4}\sum_{h\in\{-1,1\}} I_T(i+k-1,\, j+l+h) + \frac{1}{4}\sum_{h\in\{-1,1\}} I_T(i+k+1,\, j+l+h)$$

$$W_{HH} = \max_{T\in\{Y,Cb,Cr\}} \max_{\substack{k=-1..1\\ l=-1..1}} \left|w(T,k,l)\right| \qquad (5)$$

On the input of this step we have two values: possible edge direction and wavelet value. Pixel interpolation type is set to 9 if wavelet value is less then predefined thresholds. Otherwise interpolation type is set equal to the number of pixel direction.

Interpolated value of pixel is calculated using information about frequency of image area and edge type. For interpolation, two types of processing are used. Edge preserving triangulation based interpolation is computed the following way. After pixel type estimation is finished pixel value is interpolated based on triangulation and pixel interpolation type. If pixel interpolation type is 9 then pixel is interpolated using simple bi-cubic kernel. If interpolation type is less then 9 then interpolation is done using the following equations.

$$C_{s,1} = y_{s,3} - 2y_{s,2} + y_{s,1}, \quad s = 1,2$$

$$C_{s,2} = y_{s,4} - 2y_{s,3} + y_{s,2}, \quad s = 1,2$$

$$P_s = y_{s,2} + \Delta_{s,1}(y_{s,3} - y_{s,2} - \frac{2}{30}(\Delta_{s,1} - 1)(5\Delta_{s,1}(C_{s,1} - C_{s,2}) + 5C_{s,2} - 7(C_{s,1} + C_{s,2}))), \quad s = 1,2$$

$$Out = \Delta_2(P_2 - P_1) + P_1 \qquad (6)$$

Table 1. Δs,1 and Δ2 determination

Γ/P	1	2	3	4	5	6	7	8
Δ1,1	δ_y	δ_x	$\lvert \delta_y - \delta_x \rvert$	$\begin{cases}\delta_x+\delta_y, \delta_x+\delta_y \le 1\\ \delta_x+\delta_y-1, \delta_x+\delta_y > 1\end{cases}$	$\frac{\delta_y-\delta_x}{2}$	$\frac{\delta_y+\delta_x}{2}$	$\frac{\delta_x-\delta_y}{2}$	$\frac{\delta_y+\delta_x}{2}$
Δ2,1	δ_y	δ_x	$1+\lvert \delta_y - \delta_x \rvert$	$\begin{cases}\delta_x+\delta_y, \delta_x+\delta_y \le 1\\ \delta_x+\delta_y-1, \delta_x+\delta_y > 1\end{cases}$	$\frac{1-\delta_x}{2}+\delta_y$	$\frac{\delta_x-1}{2}+\delta_y$	$\frac{1-\delta_y}{2}+\delta_x$	$\frac{\delta_y-1}{2}+\delta_x$
Δ2	δ_x	δ_y	$\begin{cases}\frac{\delta_x}{1+\delta_y-\delta_x}, \delta_x > \delta_y\\ \frac{\delta_y}{1+\delta_x-\delta_y}, \delta_x \le \delta_y\end{cases}$	$\begin{cases}\frac{\delta_y}{\delta_y+\delta_x}, \delta_x+\delta_y \le 1\\ \frac{1-\delta_x}{2-\delta_x-\delta_y}, \delta_x+\delta_y > 1\end{cases}$	δ_x	δ_x	δ_y	δ_y

Where $\Delta_{s,1}$ and Δ_2 are determined using the Table 1. Let δ_x be the distance from current pixel to the nearest left column of source image, where the distance between two nearest columns of source image is assumed to be 1. Let δ_y be the distance from current pixel to the nearest upper row of source image, where the distance between two nearest rows of source image is assumed to be 1. $y_{s,k}$ are estimate using the following equation, where $I(i,j)$ – nearest left top pixel of source image to the interpolated pixel:

$$y_{s,k} = I(i+t_{s,k},\, j+p_{s,k}),\ s=1,2,\ k=1,..,4, \qquad (7)$$

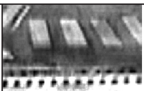
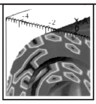
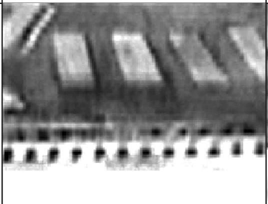
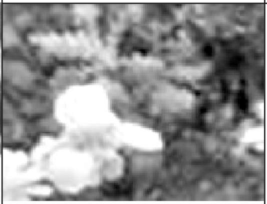
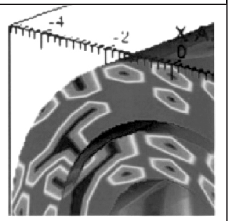
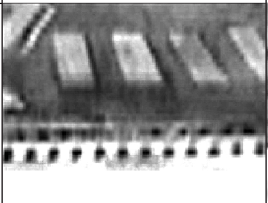
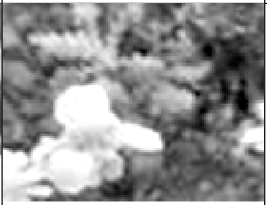
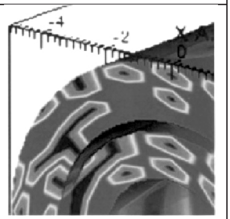
where parameters $t_{s,k}$ and $p_{s,k}$ are assumed using the Table 2. So *Out* is the out put pixel of up-sampling method.

Table 2. $t_{s,k}$ and $p_{s,k}$ determination, where 3.1 is case when $\delta_y < \delta_x$, 3.2 is case when $\delta_y \geq \delta_x$, 4.1 is case when $\delta_y + \delta_x \leq 1$, and 4.2 is case when $\delta_y + \delta_x > 1$

T/P	1	2	3.1	3.2	4.1	4.2	5	6	7	8	T/P	1	2	3.1	3.2	4.1	4.2	5	6	7	8
$t_{1,1}$	0	-1	-1	0	-1	1	0	0	-1	-1	$p_{1,1}$	-1	0	0	-1	0	-1	-1	-1	0	0
$t_{1,2}$	0	0	0	0	0	1	0	0	0	0	$p_{1,2}$	0	0	0	0	0	0	0	0	0	0
$t_{1,3}$	0	1	1	0	1	1	0	0	1	1	$p_{1,3}$	1	0	0	1	0	1	1	1	0	0
$t_{1,4}$	0	2	2	0	2	1	0	0	2	2	$p_{1,4}$	2	0	0	2	0	2	2	2	0	0
$t_{2,1}$	1	-1	1	-1	0	-1	1	1	-1	-1	$p_{2,1}$	-1	1	-1	1	-1	1	-1	-1	1	1
$t_{2,2}$	1	0	1	0	0	0	1	1	0	0	$p_{2,2}$	0	1	0	1	0	1	0	0	1	1
$t_{2,3}$	1	1	1	1	0	1	1	1	1	1	$p_{2,3}$									1	1
$t_{2,4}$	1	2	1	2	0	2	1	1	2	2	$p_{2,4}$	2	1	2	1	2	1	2	2	1	1

Suggested interpolation method is optimized for scientific data visualization system, but can be also used for different type of processing (Fig.2).

Table 3. Examples of image interpolation

Original			
Lanzcos based, x2			
Suggested, x2			

3 Image and video enhancement method

Image and video enhancement includes lots of processing method. In this paper it is described a method of detection and transformation of jagged edges in the image. Jagged edges or stair-step effect is one of the artefacts frequently appear on the rendered images.

Traditional edge enhancement algorithms are based on edges detection and smoothing. In [10] it is suggested a method of jagged edges detection and smoothing in a graphic display. Method [11] proposes a digital image luminance signal Y. The algorithm detects edge areas in the image line. Then pixel colors in these areas are estimated to make them smoother. Algorithm [12] includes an edge smoother for determining whether the detected edge is to be smoothed or rounded, or not.

It this paper it is suggested a new algorithm for jagged edges removal. Suggested algorithm works according to the following scheme (Fig. 2).

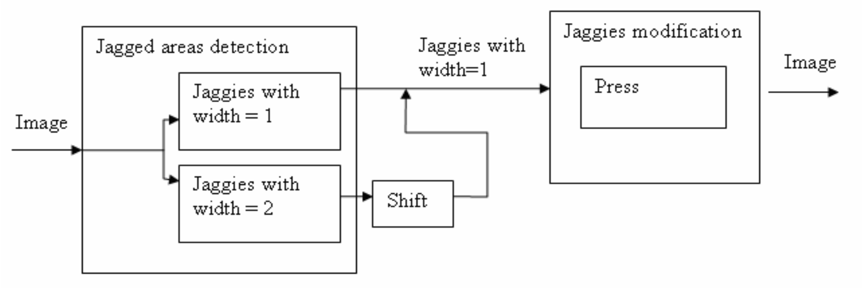


Fig.2. Flow chart of the main parts of the jaggies removal algorithm

There are two types of jaggies: horizontal and vertical. Will be described only the algorithm of detection and modification of horizontal jaggies. Horizontal jaggies can be divided into two groups. The pattern from the first group is shown in the Fig 3.a. Each line in the pattern contains jaggies sequence of colors and is shifted relatively to the next line. The other type of jaggies is shown in the Fig. 3.b. This type is characterized by two line width of the jag. Jagged pattern consists of four areas (Fig 3.c). Areas 1 and 4 are one-color. And areas 2 and 3 are jagged color sequences. Areas can be swapped left-right and up-down.

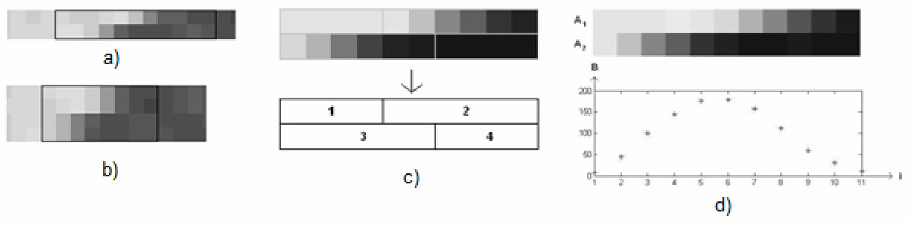


Fig.3. a) Example of jagged areas with width=1, b) Example of jagged areas with width=2, c) Jagged pattern structure, d) A plot of B(i).

Algorithm analyses two adjacent lines of an image and finds the parts of these lines where the colors of adjacent pixels are in the defined ratio. Let B be the vector of absolute value of the difference between top and bottom pixel lines (Fig 3.d). It can be seen that difference between two lines has a maximum in the middle of the pattern and reduces toward the ends. Let call this form of a plot “hill”. Jagged areas are represented as hills with slight slopes. We should localize symmetric hills on the plot of vector B. At first algorithm should detect monotonous increase and monotonous decrease parts of vector B. Algorithm contains that lengths of monotonous parts are more than 3.Then for each monotonous increase part P we check if there is a monotonous decrease part Q almost after it. “Almost after” means that the monotonous decrease part begins in the last element of part P or in the next element after last element of part P. If Q exists then we should check that difference between lengths of P and Q is not very big. We can use the following equation to do that:

$$0.5 < \frac{length\ P}{length\ Q} < 2 \qquad (8)$$

“Hill” form in the vector B is a necessary condition but not a sufficient one. We should check that color distribution in the pattern has one of the four possible schemes for jagged area (Fig..4).

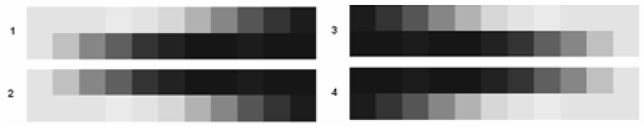


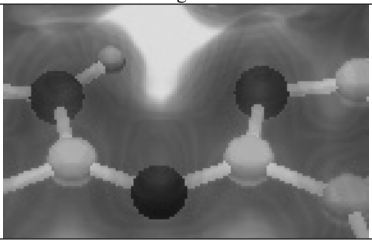
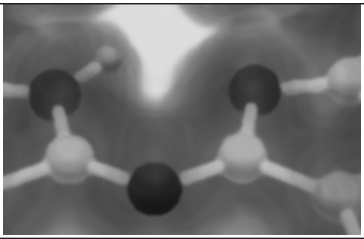
Fig.4 Color schemes of jagged areas

For jaggies modification it is suggested an algorithm based on linear press procedure. For each jagged pattern a new color of pixel is calculated by multiplication of input pixel value by linear press coefficients μ_i and η_i , where i is a number of current pixel in the pattern. Coefficients μ_i and η_i are calculated according to the following equations:

$$\mu_i = \begin{cases} 0 & ,i \in \left[1; L_1 + \frac{L_2}{4}\right] \\ \frac{4i - 4L_1 - L_2 + 2}{2L_2} & ,i \in \left(L_1 + \frac{L_2}{4}; L_1 + \frac{3L_2}{4}\right] \\ 1 & ,i \in \left(L_1 + \frac{3L_2}{4}; L_1 + L_2\right] \end{cases}; \quad \eta_i = \begin{cases} 0 & ,i \in \left[1; \frac{L_3}{4}\right] \\ \frac{4i - L_3 + 2}{2L_3}, & i \in \left(\frac{L_3}{4}; \frac{3L_3}{4}\right] \\ 1 & ,i \in \left(\frac{3L_3}{4}; L_3 + L_4\right] \end{cases} \quad (9)$$

Where the values of L$_1$, L$_2$, L$_3$, L$_4$ are depend on jagged pattern direction. For the jagged patterns of directions 1 and 4 new pixel values are calculated as follows: $I_{1,i} = \mu_i A_{1,1} + (1-\mu_i) A_{2,L_3+L_4}$ $\quad I_{2,i} = \eta_i A_{1,1} + (1-\eta_i) A_{2,L_3+L_4}$. L$_1$, L$_2$, L$_3$, L$_4$ – are the numbers of pixels in areas 1, 2, 3 and 4 of the jagged pattern accordingly. For the jagged patterns of directions 2 and 3 new pixel values are calculated as follows: $I_{1,i} = \eta_i A_{2,1} + (1-\eta_i) A_{1,L_1+L_2}$ $\quad I_{2,i} = \mu_i A_{2,1} + (1-\mu_i) A_{1,L_1+L_2}$ L$_1$, L$_2$, L$_3$, L$_4$ – are the numbers of pixels in areas 3, 4, 1 and 2 of the jagged pattern accordingly.

Table 4. Example of application of edge enhancement method

original	enhanced
	

4 Video compression method

Compression method is used for decreasing size of video sequence that is transmitted to the local workstation. Video compression part performs compression of video in different formats. Compression algorithm is based on block encoding and uses fast method of block complexity estimation for distributing processing blocks between processors using the information about block processing time that depends on block complexity. That is used for balance loading. In suggested compression algorithm bit-rate is not fixed, compression ration in this case is between 4 and 30 times. Block complexity is estimated using the information of spatial block structure. Set of parameters is calculated to estimate the complexity value. First is average value and values of dispersion that are calculated by the following equations:

$$\bar{x} = \frac{1}{k}\sum_{i=1}^{k} x_i, \quad \sigma = \frac{1}{k}\sum_{i=1}^{k}(x - x_i)^2, \quad a = \bar{x} - \sigma \times \sqrt{\frac{q}{k-q}}, \quad b = \bar{x} + \sigma \times \sqrt{\frac{k-q}{q}} \tag{10}$$

Where k is the number of pixels in block and q is the number of pixels that are more then average value. *a* and *b* are the two values that are optimal for block color representation. Also in the set of parameters it is included the values about edges in the block. This information can be obtained by calculation values according equations (1)-(4). Additionally it is checked if current block belongs to “chess” area. So it is calculated the following values for all three channels:

$$A_T = \frac{1}{4}\sum_{\substack{h\in\{-1,1\}\\ g\in\{-1,1\}}} \left| I_T(i,j) - I_T(i+h, j+g) \right|, \tag{11}$$

where T is a channel, $T \in \{Y, Cb, Cr\}$, *I(i,j)* – is a center pixel of a block.

$$B_T = \frac{1}{4}(\left|I_T(i, j-1) - I_T(i-1, j)\right| + \left|I_T(i-1, j) - I_T(i, j+1)\right| + \left|I_T(i, j+1) - I_T(i+1, j)\right| + \left|I_T(i+1, j) - I_T(i, j-1)\right|) \tag{12}$$

where T is a channel, $T \in \{Y, Cb, Cr\}$, *I(i,j)* – is a center pixel of a block.

Then if A_T < CHESS_TR and B_T < CHESS_TR, where T is at least one of the channels and CHESS_TR is a threshold for very frequency areas, and $\left|I_T(i,j) - I_T(i, j-1)\right|$ > CHESS_UP_TR then this pixel is considered to belong to chess area. Then for block complexity estimation it is calculated an integral value of the values of estimated parameters and then this integrated value is used for distribution of blocks to the processing unit to optimize the loading of processors. Distribution scheme is described in work [13].

After complexity estimation for each block it is calculated optimal palette for block representation. Number of colors in palette depends on block complexity. Palette is estimated using iterative method of deleting zero intervals from block histogram. On each iteration it is selected the longest zero interval and colors with zero values and excluded from histogram. Process stops when there are no zero intervals or number of iterations is more then number of colors in palette. Colors of palette are estimated as middle values of histogram intervals that were not excluded. The process of palette estimation is illustrated on the Fig.5.

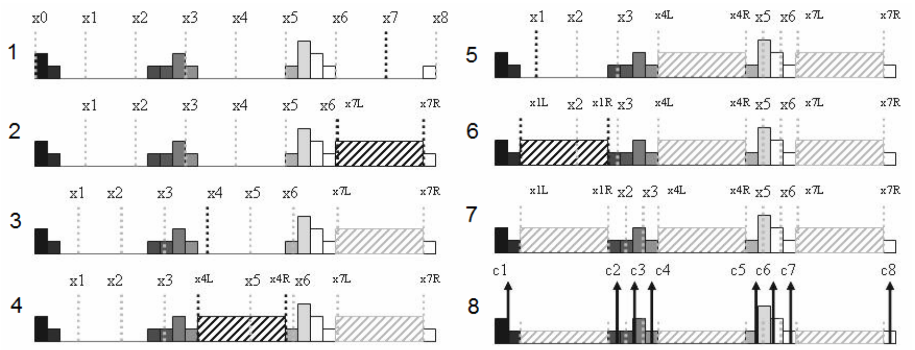

1
x0 x1 x2 x3 x4 x5 x6 x7 x8
2
x1 x2 x3 x4 x5 x6 x7L x7R
3
x1 x2 x3 x4 x5 x6 x7L x7R
4
x1 x2 x3 x4L x5 x4R x6 x7L x7R
5
x1 x2 x3 x4L x4R x5 x6 x7L x7R
6
x1L x2 x1R x3 x4L x4R x5 x6 x7L x7R
7
x1L x1R x2 x3 x4L x4R x5 x6 x7L x7R
8
c1 c2 c3 c4 c5 c6 c7 c8

Fig.5 Steps of eight color palette estimation.

5 Main results

Suggested image and video processing methods were applied to MSU_VIZ visualization system for large scale data in high performance computing. Different test of effectiveness were produced. Algorithms have low computational complexity and good visual quality of produced video content. Fig.6 illustrates some quantitative results. Fig.6.a shows the decrease of processing time in depends of the size of rendering image. We can see strong time reduction for all number of processors. On the Fig.6.b it is shown the depends of speed of compression process for different number of processors. Speed up is almost linear so the algorithm has good scalability. Such results are reached using the algorithm of block complexity estimation and optimal blocks distribution through processors.

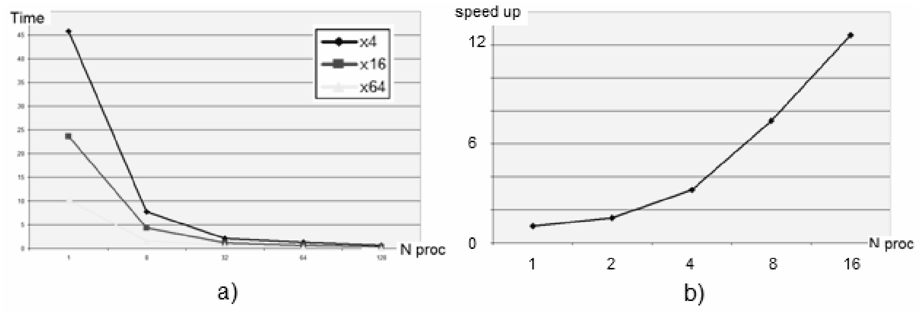


Fig.5 a) Dependence of rendering time from interpolation ratio; b) Speed up of compression process with the scheme of optimal blocks distribution through the processors.

5 Conclusion

This paper present three video processing methods that can be used for large scale data visualization system for high performance computing. It is suggested effective method of video up-sapling that is used for decreasing the size of frame that is produced during rendering. Up-sampling method is based on triangulation and wavelets. Also it is suggested a method of video rostprocessing for visual quality enhancement. Method is based on reduction of jagged edges on the image. One more method is video compression method which is based on block oriented palette estimation algorithm that is used for video encoding before transmitting to the local work station. Suggested algorithms have low computational complexity and produces good visual results. Methods were applied to SU_VIZ visualization system for large scale data processing.

This work is done with supporting of RFFI grant № 08-07-12081.

References

1. Dzhosan, O.: Scientific Data Visualization for High Performance Parallel Applications. Pros. conf. PAVT2009, Nijniy Novgorod, Russia, 2009
2. Dutra, M., Rodrigues, P., Giraldi, G., Schulze, B.: Distributed visualization using VTK in Grid Environments. Pros. conf. Seventh IEEE International Symposium on Cluster Computing and Grid (CCGrig'07), 2007
3. Moreland, K., Avila, A., Fisk, L.A.: Prallel Unstructured Volume Rendering in ParaView. Proceedings of IS&T SPIE Visualization and Data Analysis 2007, San Jose, 2007
4. Childs, H., Duchaineau, M., Ma, K.L.: A Scalable, Hybrid Scheme for Volume Rendering Massive Data Sets. Proceedings of Eurographics Symposium on Parallel Graphics and Visualization 2006, Braga, Portugal, 2006
5. Peterka, T., Yu, H., Ross, R., Ma, K.-L.: Parallel Volume Rendering on the IBM Blue Gene/P, pros. conf. Eurographics Symposium on Parallel Graphics and Visualization, 2008
6. Installation of leading-edge data analytics, visualization set for world's fastest open science supercomputer, http://www.anl.gov/Media_Center/News/2008/news080722.html

7. Blu, T., Thvenaz, T., Unser M.: Generalized interpolation: higher quality at no additional cost. Proc. Int. Conf. Image Process.1999, vol. III, pp. 667-671
8. Blu, T., Thvenaz, P.: Interpolation revisited. In: IEEE Trans. On Medical Imaging, vol. 19, No 7, 1999, pp 739-758
9. Yu, X., Morse, B., Sederberg, T.: Image Reconstruction Using Data-Dependent Triangulation. IEEE Computer Graphics and Applications, vol. 21 No. 3, 2001, pp. 62-68
10. John, F.: Method and apparatus for smoothing jagged edges in a graphics display. USA Patent 5293579, 1994
11. Park,Y.J.: Method and circuit for correcting image edges. USA Patent 5151787, 1992
12. Tults, Y.: Graphical on-screen display system. USA Patent 6339451, 2002
13. Dzhosan, O.,Mishourovsky, M.: Analysis of methods for visually lossless colour image compression with low complexity. Proceedings of “Television: Images Transmitting and Processing” Conference, Russia, St.-Petersburg, 2008

Sequential Probabilistic Models for ECG Segmentation

Yuliyan Velchev[1],

[1] Faculty of Telecommunications at Technical University of Sofia, 8 Kl. Ohridski blvd,
Sofia 1000, Bulgaria,
julian_velchev@abv.bg

Abstract. In this paper an application of several sequential probabilistic models for ECG segmentation purpose is present. The observations for the probabilistic models are vectors, which are coefficients from appropriate wavelet transform applied on the ECG signal. The final choice of the parameters of the wavelet transform (wavelet function, scales etc.) is based on detection performance optimization by using ROC diagrams. The proposed approach for segmentation performance evaluation is to use a synthetic ECG signal with additive white Gaussian noise in terms to model the muscle artifacts. This approach also eliminates the subjective criterion for onset and offset annotation of a given ECG component in the training procedures for the probabilistic models.

Keywords: ECG segmentation, Sequential probabilistic models, Hidden Markov model, Conditional Random Field.

1 Introduction

The electrocardiogram (ECG) is a record of time varying bioelectric potentials generated by electrical activity of the heart. Normal human ECG consists of components named waves, complexes and segments (Fig. 1).

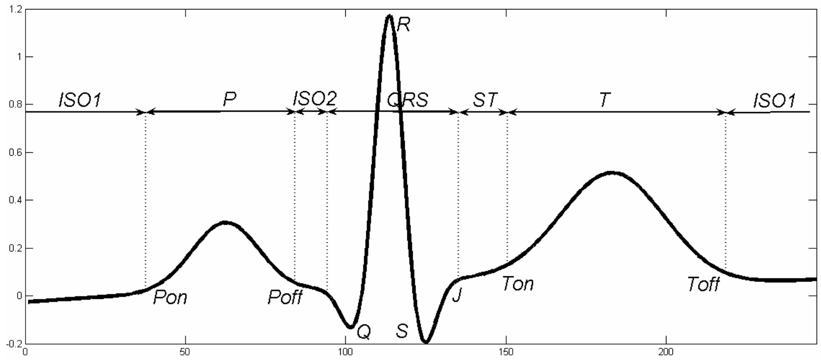


Fig. 2. The standard components (waves, complexes, segments) of a typical human ECG waveform.

Sequential depolarization of the heart's right and left atrium reflects in ECG as P wave. The QRS complex corresponds to right and left ventricle depolarization. Ventricular repolarization forms the T wave in ECG. The regions between waves (complexes, segments) are defined as isoelectric line or baseline. Manual ECG analysis is considered to be hard and impractical, especially for long-term ECG recordings. By this reason, improvement of the methods and algorithms in the field of automatic ECG analysis gives significant advantages for medical staff for pathology events detection [11]. The signal segmentation (onset and offset determination of the mentioned above ECG components) is the most important task in automatic ECG analysis. The contemporary concept for this problem is to incorporate a suitable probabilistic model in terms to assign each sample of the ECG signal to a given ECG component (wave, complex or segment). Some advanced probabilistic approaches for ECG segmentation are present in [1], [3]. Typically they are based on hidden Markov models (HMM) or hidden semi-Markov models (HSMM). The main goals of this paper are: improvement of existing HMM-based methods for ECG segmentation, testing and evaluating the conditional random field (CRF) for ECG segmentation purpose and comparative analysis of mentioned sequential probabilistic models.

The remaining of this paper is organized as follows. In section 2 the features selection for probabilistic models is given. These features (observation vectors) are composed by using wavelet transform (WT) with optimal parameters, which are found by maximization of segmentation performance. Section 3 proposes an architecture of the analyzed sequential probabilistic models (GRF, HMM) and introduces the application of CRF for ECG delineation. In section 4 a comparative analysis of the effectiveness for the different probabilistic models is taking place.

2 Wavelet Transform Features

One of the HMM requirements is the statistical independence between consecutive observations. Since ECG signals don't satisfy this assumption, the HMM can perform better when working with observation vectors resulting from appropriate time-frequency signal transform (wavelet transform is used in this approach). The coefficients from WT W serve as similarity measure between signal and a wavelet function at different scales s and time shifts τ as parameters [6]:

$$W_x(s,\tau) = \langle x, \psi_{s,\tau} \rangle = \frac{1}{\sqrt{s}} \int_{-\infty}^{\infty} x(t) \psi^* \left(\frac{t-\tau}{s} \right) dt, \qquad s \in \mathrm{R}^+, \tau \in \mathrm{R}, \tag{1}$$

where $x(t)$ is the ECG signal and ψ is the wavelet function

The discrete wavelet transform (DWT) is not suitable for this purpose, since the transform is not time-invariant and the resulting coefficients are decimated, so it is impossible to build observation vectors associated to a given signal sample. The continuous wavelet transform (CWT) doesn't have these limitations, but it should be used with reduced set of scales, because this transform has large amount of information redundancy. In the case, with reduced number of scales, the perfect signal reconstruction is not possible. The stationary wavelet transform (SWT) possesses compact support (perfect signal reconstruction is possible), but the number of available wavelet functions is limited. As the perfect signal reconstruction is not necessary in this case, the CWT with dyadic scales is used to build the observation vectors. The wavelet function ψ should satisfy several requirements: maximum similarity between ψ and ECG components; minimal effective width; it is also desirable for ψ to be symmetric in order to avoid complicated phase corrections. The wavelet functions, which can fulfill these requirements, are: second derivative of the Gaussian function (Mexican hat), Morlet and the last three wavelets from Coiflet family (Table 1).

Table 1. Potentially suitable wavelet functions for ECG segmentation purpose and their most important characteristics.

Wavelet Function	Definition	Effective Width	Symmetry (Anti-symmetry)
Gauss2	$\psi(t)=\frac{1}{\sqrt{2\pi\sigma^3}}\left(1-\frac{t^2}{\sigma^2}\right)e^{-t^2/2\sigma^2}$	[-5, 5]	Symmetric
Coiflet3	Coiflet coefficients	17	Approx. symmetric
Coiflet4	Coiflet coefficients	23	Approx. symmetric
Coiflet5	Coiflet coefficients	29	Approx. symmetric
Morlet	$\psi(t)=e^{-t^2/2}\cos(5t)$	[-4, 4]	Symmetric

The final choice is made by segmentation performance comparison of a HMM combined with different wavelet functions. The segmentation performance criterion is receiver operating characteristics (ROC) plots (discussed in detail in section 4).

The suitability of the mentioned above wavelet functions for the given purpose is evaluated by testing a HMM with single Gaussian model core (HMM-SGM). Its observations vectors are built form WT coefficients for each tested wavelet function superlatively. As can be seen from Fig. 2, and Fig. 3, the Mexican hat significantly outperforms the other wavelets. The Mexican hat wavelet doesn't support SWT, so it is impossible to reconstruct the original ECG signal.

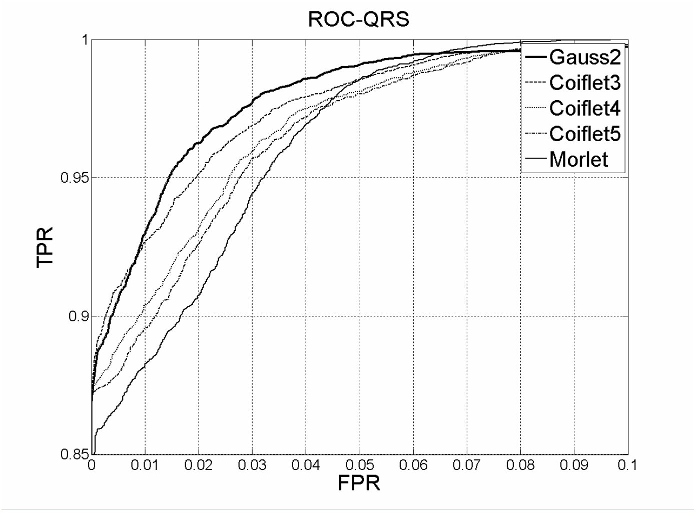


Fig. 2. Detection performance for the QRS complexes by using HMM-SGM with different wavelet functions.

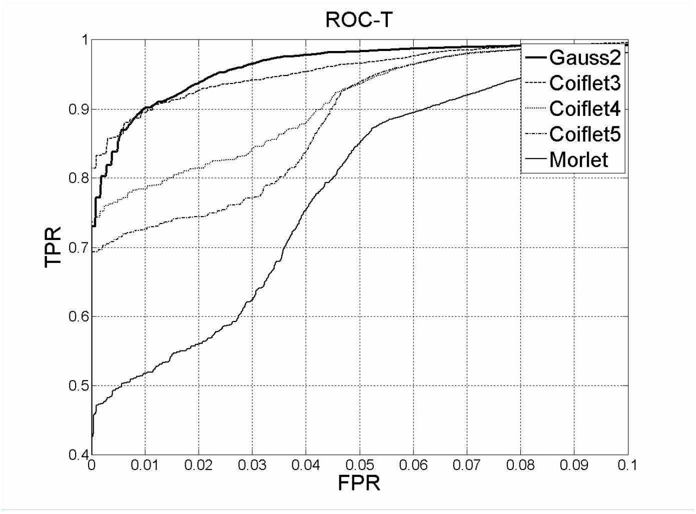


Fig. 3. Detection performance for the T waves by using HMM-SGM with different wavelet functions.

Also by considering the ROC analysis the optimal dyadic wavelet scales are determined to be form 2 up to 64 (when working with scales from 1 up to 64, the performance is increasing insignificantly). So the observation vectors (features) **f** are built as follows:

$$\mathbf{f}_{O\equiv\tau} = \left[W_x(2,\tau), W_x(4,\tau), W_x(8,\tau),\right.$$
$$\left.W_x(16,\tau), W_x(32,\tau), W_x(64,\tau)\right]^T \tag{2}$$

3 Sequential Probabilistic Models for ECG segmentation

3.1 Conditional Random Field

The conditional random field (CRF) are undirected graphical model for sequential data classification [4], [5], [8], [9]. CRF directly models the conditional probability distribution of the hidden variables (the set of the possible states). CRF is discriminative model and it generally supports random dependencies between the observations. There are two categories of CRF: linear chain CRF and CRF with complicated dependencies between states and observations. For a linear chain CRF the following conditional probability is valid:

$$P(\mathbf{S}\,|\,\mathbf{O}) = \frac{1}{Z}\prod_{i=1}^{N}\exp\left(\sum_{k=1}^{K}\lambda_k f_k(\mathbf{O}, i, S_{i-1}, S_i)\right), \tag{3}$$

where f_k are random features, which can overlap and Z is the partition function. According to [4] the asymptotic error of the CRF is smaller than HMM, but this error is reached significantly slower than HMM. By this reason the detection of some short-time ECG segments could be problematic.

The CRF training is quite complicated. Usually it employs the quasi-Newton (L-BFGS) method. The CRF is "bias problem" free (the "bias problem" is very typical for hidden Markov models).

3.2 Hidden Markov Model

The HMM is a probabilistic model, which describes the statistical dependency between an observation sequence O and sequence of hidden states S [7]. A HMM λ is denoted with:

$$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{q}), \tag{4}$$

where $\mathbf{A}$ is transition probability matrix, $\mathbf{B}$ is observation probability distribution and $\mathbf{q}$ is initial distribution vector. The model used in this project is first order Markov process (Markov chain):

$$P(s_{t+1} \mid s_t, s_{t-1}, s_{t-2}, \ldots) = P(s_{t+1} \mid s_t), \quad \textbf{(5)}$$

where s_t is the state (ECG component) for time t. The ECG segmentation task is to find the optimal state sequence S' given the HMM parameters λ and observation sequence O:

$$\begin{aligned} S' &= \underset{S}{argmax}\{P(S \mid O, \lambda)\} = \\ &= \underset{S}{argmax}\left\{\frac{p(S, O \mid \lambda)}{p(O \mid \lambda)}\right\} = \\ &= \underset{S}{argmax}\{p(S, O \mid \lambda)\} \end{aligned} \quad \textbf{(6)}$$

The optimal state sequence S' in (3) is determined as the most probable state sequence given the observation. The final result is achieved using Bayes' rule. An effective way to calculate (3) is to use the Viterbi algorithm [10].

HMM learning (finding the HMM parameters λ) is a task, which can be done using supervised or unsupervised techniques. Supervised learning is related to maximum likelihood:

$$\lambda' = \underset{\lambda}{argmax}\{p(\mathbf{O} \mid \lambda)\}, \quad \textbf{(7)}$$

where λ' represents HMM parameters, which maximize the probability and $\mathbf{O} = \{O_1, O_2, \ldots, O_N\}$ are the observation sequences.

Unsupervised learning generally employs the Expectation Maximization (EM) algorithm. It is only possible when more than one Gaussian component in the mixture is used i.e. HMM with Gaussian mixture model (HMM-GMM). It has been proved through experiments that the EM algorithm can be applied successfully only to "refine" the already learned parameters that are obtained from supervised training.

3.3 Model's architecture

The proposed architecture for the sequential probabilistic models is according to the regions of interest in ECG signals (standard waves, complexes or segments) (Fig. 4). Sometimes the region between QRS complex and T wave (JT segment) is hard to be distinguished, so the proposed model allows direct transition from QRS complex to T wave. Since the model is trained with ECG signals from healthy persons, all remaining transitions are according to Fig. 1.

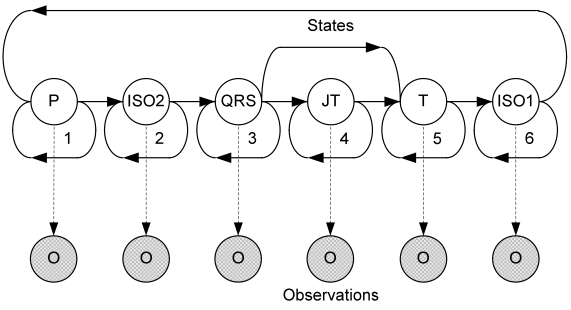


Fig. 4. Architecture of the sequential probabilistic models for ECG segmentation.

4 Experimental Results

4.1 Criterion for Segmentation Performance

The criterion for segmentation performance for a given probabilistic model is according to receiver operating characteristics (ROC) plot. It represents the dependency *TPR = f(FPR)*, where *TPR* is true positive rate and *FPR* is false positive rate:

$$TPR = \frac{TP}{TP + FN} \tag{8}$$

$$FPR = \frac{FP}{FP + TN}, \tag{9}$$

where *FP* is the number of false positive detections, *TN* is the number of true negative detections, *TP* is the number of true positive detections and *FN* represents the number of false negative detections. The ROC plot represents binary classification performance i.e. it is evaluated for each ECG component, which is the first class and all other remaining components form the second.

An additional criterion for segmentation performance is the absolute error between annotated and detected position of a characteristic point in the ECG signal (onset or offset of a given ECG component):

$$e_p = t_{p,a} - t_{p,d}, \tag{10}$$

where the index p denotes a characteristic point in ECG signal, $t_{p,a}$ is expert annotated position of the characteristic point and $t_{p,d}$ is the detected position by the probabilistic model for the same point.

4.2 Comparative analysis

The segmentation performance comparison for HMM-SGM, HMM-GMM (8 mixtures) and CRF is given in Fig. 5.

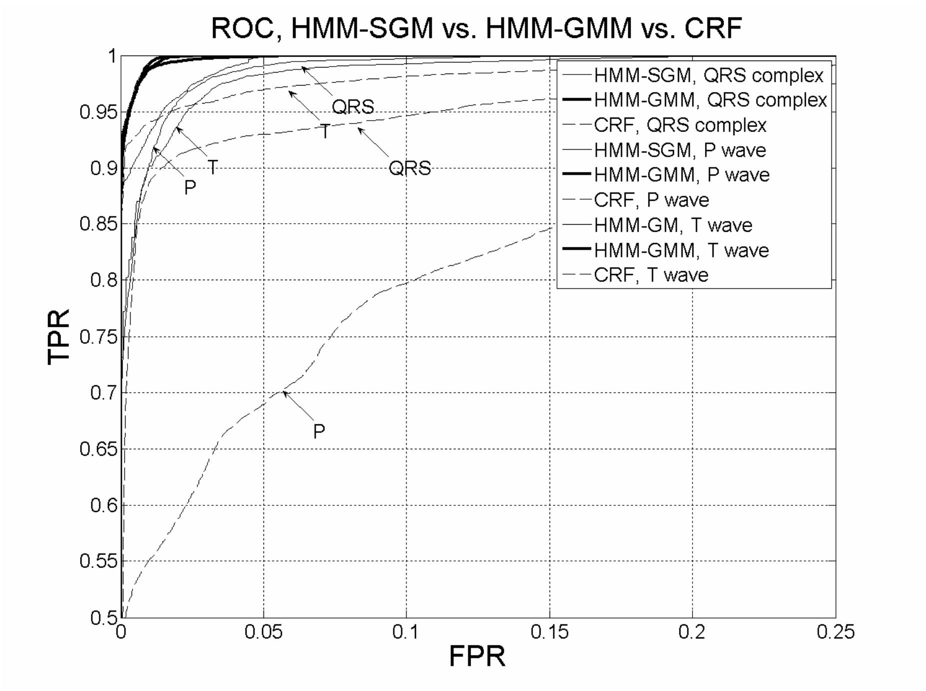


Fig. 5. Segmentation performance for HMM-SGM, HMM-GMM (8 mixtures) and CRF.

The models have been trained with synthetic ECG signal with duration of about 10 s, sampling rate of 128 Hz and with simulated variable heart rate (from 50 bpm up to 120 bpm). The models have been tested with same synthetic ECG signal, but with increased duration (about 5 min). As can be seen (Fig. 5), the slowly reached asymptotic error in CRF significantly reduces its performance for segments with short duration (such as P waves). The best results are achieved when using HMM-GMM.

The HMM-GMM is evaluated in terms of absolute error for onsets of all ECG components (Table 2.). When dealing with noise free ECG signal, the mean and standard deviation of the error is comparable to sampling interval.

Table 2. Statistical characteristics (mean and standard deviation) of the absolute error of the detected characteristic points by using HMM-GMM (8 mixtures) on synthetic ECG signal with/without AWGN.

	Pon	ISO1on	QRSon	JTon	Ton	ISO2on
	μ_e ,ms σ_e , ms	μ_e ,ms σ_e , ms	μ_e ,ms σ_e , ms	μ_e ,ms σ_e , ms	μ_e ,ms σ_e , ms	μ_e ,ms σ_e , ms
Noise free ECG	-3,8 6,1	5,8 4,2	-11,1 5,1	4,0 5,0	-6,9 7,2	4,6 6,6
ECG with AWGN – SNR 10 dB	7,5 48,1	-4,8 91,6	16,2 7,1	0,4 7,1	4,4 12,3	-4,6 18,5

In Fig. 6 can be seen a sample segmentation result of ECG with additive white Gaussian noise (AWGN), when using HMM-GMM (8 mixtures).

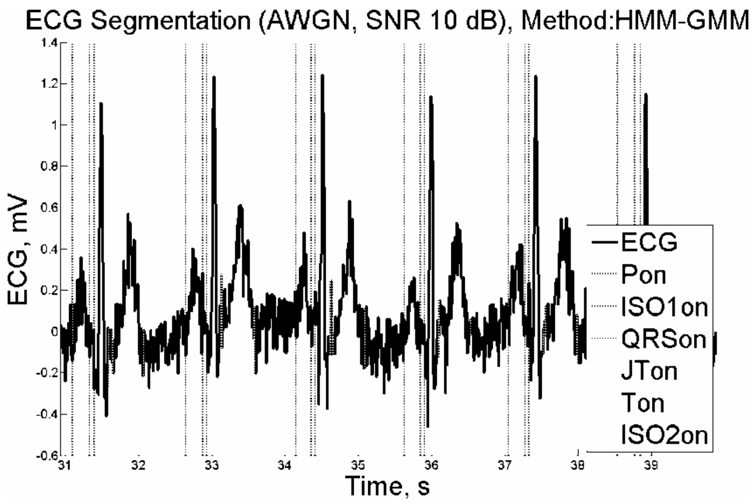


Fig. 6. ECG segmentation result for HMM-GMM with AWGN influence (10 dB).

All the sequential probabilistic models are based on Conditional Random Field Toolbox for Matlab [2].

5 Conclusion

In this paper the ECG segmentation algorithms with hidden Markov models and conditional random field were present. The observation vectors for the models are composed from wavelet transformation coefficients. The proposed algorithms are evaluated with synthetic ECG signal with/without Gaussian noise. Also a comparative analysis based on receiver operating characteristics was made. According to the achieved results, as future work it is intended to improve the performance of HMM-GMM algorithm and to develop methods and algorithms for ECG analysis, applied on already segmented signal.

Acknowledgement This work was sponsored by the Ministry of Education and Science of Bulgaria, National Foundation "Science and Research", contract BY-TH-202/2006: "Multimodal biometric analysis: methods and algorithms".

References

1. Andreão R., Boudy J.: Combining Wavelet Transform and Hidden Markov Models for ECG Segmentation. EURASIP Journal on Advances in Signal Processing, vol. 2007 (1), pp. 95--95, (2007)
2. Conditional Random Field (CRF) Toolbox for Matlab, http://www.cs.ubc.ca/~murphyk/Software/CRF/crf.html
3. Hughes N.P., Tarassenko L., Roberts S.J.: Markov Models for Automated ECG Interval Analysis. In Advances in Neural Information Processing Systems, vol. 16, NIPS, (2003)
4. Lafferty J., McCallum A., Pereira F.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282--289, (2001)
5. Liu Y.: Conditional Graphical Models for Protein Structure Prediction. PHD Thesis, Carnegie Mellon University (2006)
6. Mallat S.: A Wavelet Tour of Signal Processing. Academic Press, 2nd edition, (1999)
7. Rabiner L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, vol. 77 (2), pp. 257--286, (1989)
8. Sutton C., McCallum, A.: An Introduction to Conditional Random Fields for Relational Learning. Introduction to Statistical Relational Learning, MIT Press (2007)
9. Quattoni A., Collins M., Darrell T.: Conditional Random Fields for Object Recognition. Advances in Neural Information Processing Systems 17 (NIPS 2004), (2005)
10. Forney G.D.: The Viterbi algorithm. Proceedings of the IEEE, vol. 61, pp. 268--278, (1973)
11. Velchev Y., Boumbarov O.: A Probabilistic ECG QRS Detector, International Conference Automatics and Informatics '08, Proceedings, Sofia, Bulgaria, vol. 2, pp. 29--33, (2008)

Using Ontologies in Building Intelligent Sensor Systems

Maria M. Nisheva-Pavlova[1]

[1] Faculty of Mathematics and Informatics, Sofia University St. Kliment Ohridski,
5, James Bourchier Blvd., 1164 Sofia, Bulgaria
marian@fmi.uni-sofia.bg

Abstract. The paper discusses the main features and design principles of intelligent sensor systems and analyzes the applicability of different types of ontologies at the Knowledge Processing level in these systems.

Keywords: Intelligent Sensor, Knowledge Processing, Ontology, Semantic Services, Semantic Web.

1 Introduction

During the last years, a new generation of sensor systems, intelligent sensor systems, have became the core of research and practical implementations in many organizations. An intelligent sensor should be *autonomous*, *adaptive* to changes in its environment and *reactive*. More precisely, intelligent sensor systems are intended to provide the following security services:

- Store, sort, associate, aggregate and report event information;
- Detect intrusions and anomalies;
- Centralize collection and archiving.

Intelligent sensor systems perform various types of computation:

- Signal conditioning (e.g., filtering);
- Signal conversion (e.g., analog to digital);
- Logic functions (e.g., triggering events);
- Data reduction (e.g., feature extraction);
- Decision making (e.g., classification).

The functionalities of intelligent sensor systems presume that they have a hierarchical structure with different abstraction layers (fig. 1):

- The ***Lower layer*** performs *Signal Processing* (conditioning, filtering, conversion, contrast enhancement etc.);
- The ***Middle layer*** performs *Information Processing* (feature extraction, sensor signal fusion and parameter tuning);
- The ***Upper layer*** performs *Knowledge Processing* (clustering, prediction, classification, and more general, decision making). The overall control of the

system and the communication with the environment are also accomplished within this layer.

The paper discusses several aspects of the applicability of various types of ontologies in order to assist some of the most important types of computation performed by intelligent sensor systems at Knowledge Processing level.

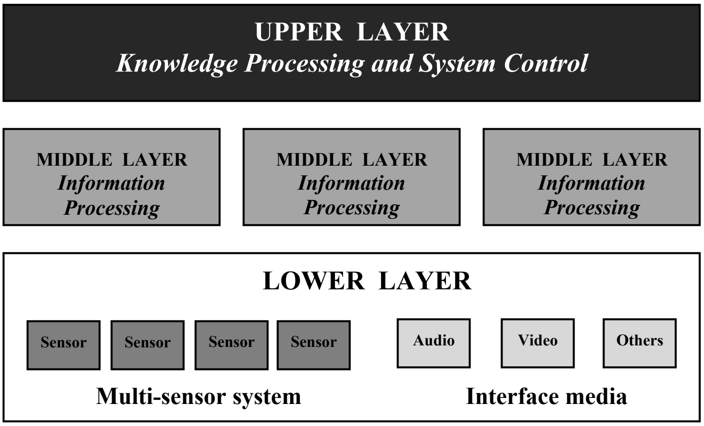

UPPER LAYER
Knowledge Processing and System Control
MIDDLE LAYER Information Processing
MIDDLE LAYER Information Processing
MIDDLE LAYER Information Processing
LOWER LAYER
Sensor
Sensor
Sensor
Sensor
Audio
Video
Others
Multi-sensor system
Interface media

Fig. 1. Hierarchical structure of intelligent sensor systems

According to the most cited definition in Computer Science, an *ontology* is an explicit specification of a conceptualization [5]. This definition was later modified and extended by Borst: “an ontology is a formal specification of a shared conceptualization” [2]. Ontologies define domain concepts and the relationships between them, and thus provide a domain language that is meaningful to both humans and machines. They are formal theories supporting knowledge sharing and reuse and in this sense form the basis of the Semantic Web. In this paper, we use a more precise notion that is fairly well captured by Guarino: “an ontology is a set of logical axioms designed to account for the intended meaning of a vocabulary” [6].

People, organizations and software systems need to communicate and share information, but due to different needs and background contexts, there can be widely varying viewpoints and assumptions regarding what essentially the subject matter is. The lack of shared understanding leads to poor communication between people and their organizations, severely limits systems interoperability and reduces the potential for reuse and sharing. Ontologies aim at solving all these problems.

On the one hand, ontologies facilitate communication and knowledge sharing by providing a unifying framework for parties with different viewpoints. On the other hand, ontologies can improve interoperation and cooperation by providing unambiguous semantics in a formal, machine-interpretable way.

2 Use of Ontologies in the Development of Semantic Services for Intelligent Sensor Systems

A significant barrier to the wider adoption of sensor systems by novice end users lies in the lack of an information framework within which one can ask what information is to be desired and how the information is to be extracted from raw and noisy sensor data. There are often more than one way to obtain a piece of information, and one has to understand the information equivalency relations over the rich variety of sensor data. An intelligent information system which supports decision making on the base of data obtained by a multi-sensor system requires a semantic information hierarchy and a set of semantic services to process and reason about sensor data, moving beyond just protocol agreement and data format conversion.

To quantify the semantic information contained in sensor data and to capture the relations between various semantic entities such as objects and events in the physical environment, we need to define an *ontology for a variety of commonly encountered sensor data*. We have also to provide a set of transformations, or *semantic services*, that can incrementally extract new semantic information from lower level data [7].

The raw sensor data can be hard to interpret by ordinary users. But, by using a sequence of these sensors, one may for example distinguish whether the passing by object is a human or a vehicle. By estimating the vehicle's speed, length and number of wheels, one may further infer whether it is a car or a bicycle. One may also distinguish a car from a bicycle by inferring the amount of metal the object has. This requires more domain knowledge of magnetic fields**.** By analogy, specific domain knowledge should be used in order to infer whether the moving human is a man, a woman or a child.

Now, if the system has access to a common ontology that defines concepts such as "mobile object", "vehicle", "car", "bicycle" and their relations, as well as properties such as "speed", "length", "direction", "metal", "pulse", and "magnetic signal", one may encapsulate sophisticated domain knowledge and signal processing algorithms into computational components, or services, that can be reused. Each service simply transforms input events into output events, adding new semantic information as necessary. For example, a **`SpeedEstimation`** service may take a sequence of pulses and produce the speed of the object [7].

Using such ontology, the end users can specify what they need to compute, say e.g. "detecting a red car passing the garage entrance" or "a man entering the room", without explicitly referring to low-level sensor signals such as camera images; the system can take care of instantiating the requisite semantic services, when wired together, will produce the required semantic events, i.e., the red car events. The system may even choose among alternate detection methods to satisfy contextual or resource constraints.

Therefore, ontologies can be used at the knowledge processing level in order to capture the information about the physical entities sensors are trying to sense and their relations as well. Having a common ontology for each application domain helps unify information presentation and permits software and information reuse.

For example, the left panel on fig. 2 describes a simple information inheritance hierarchy for the vehicle detection example discussed above. Each class represents a concept in the ontology and the typical properties that the conceptual object may have are explicitly defined. For example, the diagram captures that cars and bicycles are sub-classes of vehicles, which are subclass of mobile objects, as well as metal objects. The classes of human beings and metal objects are described as disjoint. Therefore, to distinguish a vehicle from a person, one can test if the object is a metal object or not, which uses the structural information defined in the ontology, or one can test if the length of the object exceeds a threshold, which exploits differences in object property values to classify objects.

In such ways the use of proper domain ontologies could significantly assist the development of semantic services for intelligent sensor systems.

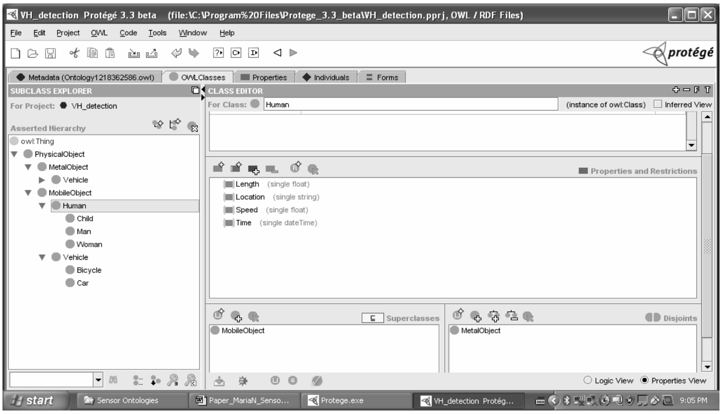


Fig. 2. Part of a vehicle/human detection ontology created using Protégé/OWL

3 Use of Ontologies in the Development of Sensor Networks

In the more complex case of building intelligent sensor networks, a special kind of semantic brokering service should be developed [9]. This service could use a general sensor ontology to describe the capabilities of the resources in the network. This ontology should include major properties such as sensor location and sensing mechanism. Each major property could be specified with low-level properties. For example, every sensor should choose at least one possible sensing mechanism to describe itself.

The service mentioned above has the task to assign a specific mission to each node of the sensor network. This might be considered as a problem of *matchmaking*: decomposition of the user-defined requirements to the functionalities of the network to a set of well-defined missions and then matching the specifications of the missions with the descriptions of the capabilities of the available network nodes (sensors and the platforms that carry them – the systems to which sensors are attached so as to get energy, protection, mobility, communication, etc.). A possible approach to the achievement of the aim could be based on the use of ontologies as formal models of the various elements that may be used with deductive reasoning mechanisms to produce matches that are logically sound. More precisely, the idea here is to develop an unified approach to the specification of the requirements of the particular missions and the capabilities provided by the means, i.e. the network elements (sensors and platforms). Thus the problem could be solved by comparing the specification of each mission against the specifications of the available means and then obtaining a set of recommended means for the mission. More precisely, a decision should be made that there is a solution that satisfies the requirements of the mission, or alternatively a ranking of candidate solutions according to their relative degree of utility should be suggested.

Matchmaking can benefit from the general properties of ontologies as far as the elements of the process are distributed or there are several viewpoints to them; additionally, the use of semantically rich specifications enable the use of specific forms of reasoning that are not available when using a syntactic approach, such as for example subsumption and disjunction.

A benefit of such an approach is the fact that it can be grounded on much existing work in representing sensors and platforms for various purposes, and that current Web standards for ontology engineering provide a suitable foundation for it.

There is already a sizeable amount of work in providing descriptive schemas for sensors, sensor platforms, and their properties, for example the OpenGeospatial Consortium (OGC, http://www.opengeospatial.org) suite of Sensor Web Enablement (SWE, http://www.opengeospatial.org/projects/groups/sensorweb) specifications. There are also a number of prototype ontologies for sensors and sensor platforms including OntoSensor [4,10], CIMA [8], the Marine Platforms Ontology [1], and an ontology for level 1 sensor fusion [3].

OntoSensor is designed as a prototype sensor knowledge repository, which is compatible with the evolving Semantic Web infrastructure [4,10]. It has been used to mark-up live data from sensors in a network to achieve efficient data fusion. The design approach of OntoSensor has been a compositional one. OntoSensor includes some concepts from SensorML (see below) and refers to ISO 19115 for definitions regarding geographic information. It has also extended the upper level of IEEE Suggested Upper Merged Ontology (SUMO, http://suo.ieee.org/SUO/SUMO/index.html) – e.g. Sensor and Platform concepts of OntoSensor extend MeasurementDevice and TransportationDevice concepts of the IEEE SUMO ontology.

SensorML (http://vast.uah.edu/SensorML) is originally designed as a standalone XML-based sensor model language for describing the geometric, dynamic, and radiometric properties of dynamic remote sensors. Recently it has become an integral part of the SWE initiative of the OGC. In the SWE approach, sensors are self-describing Web components, which are discoverable and accessible in real time by Web services. Also, sensor information and their observations are made accessible in real-time through Web services, thus enabling sensor systems to find phenomena of immediate interest autonomously. SensorMLs role in SWE is to provide XML

Schemas for describing sensor systems and processes, information needed for the discovery of sensors, location of sensor observations, and processing of low-level sensor observations.

4 Conclusion

In this paper we have argued that the use of different types of ontologies could effectively aid the development and improve the functioning of intelligent sensor systems. The success of the development of semantic services for intelligent sensor systems depends on the efforts to define ontologies in a number of domains. Encouraging in this sense is the existence of a significant number of free repositories with already created ontologies, e.g. the Protégé Ontologies Library (http://protegewiki.stanford.edu/index.php/ Protege_Ontology_Library).

The semantic representation of sensor networks nodes and data is an exciting vision that maximizes the quality of search engines. Increasing the precision and recall rates is a necessary prerequisite for automatic search, retrieval, and processing of sensor data. A further step in this direction should be the definition of a universal ontology for describing concepts and relationships for the sensor networks nodes and data.

Acknowledgments. This research is supported by Project VU-MI-204/2006 “Intelligent Sensor Systems for Security Enhancement” funded by the Bulgarian National Science Fund.

References

1. Bermudez, L., J. Graybeal, R. Arko. A Marine Platforms Ontology: Experiences and Lessons. In: Proceedings of the 2006 Workshop on Semantic Sensor Networks, Athens GA, USA, 2006.
2. Borst, W. Engineering Software Library Systems. PhD thesis, Twente University, 1997.
3. Ceruti, M. Ontology for Level-One Sensor Fusion and Knowledge Discovery. In: Proceedings of the 2004 Workshop on Knowledge Discovery and Ontologies, Pisa, Italy, 2004. http://olp.dfki.de/pkdd04/ceruti-final.pdf, last accessed on February 17, 2009.
4. Goodwin, C., D. Russomanno. An Ontology-Based Sensor Network Prototype Environment. In: Proceedings of the 5th International Conference on Information Processing in Sensor Networks, Nashville TN, USA, 2006, pp. 1–2. http://www.cs.virginia.edu/~ipsn06/WIP/ goodwin_1568983444.pdf, last accessed on February 17, 2009.
5. Gruber, T. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human-Computer Studies, Vol. 43, 1995, pp. 907-928.
6. Guarino, N. Formal Ontologies and Information Systems. In: Proceedings of the 1st International Conference on Formal Ontologies in Information Systems (FOIS-98). IOS Press, June 1998, pp. 3–15.
7. Liu, J., F. Zhao. Towards Semantic Services for Sensor-Rich Information Systems. Proceedings of the 2nd IEEE/CreateNet International Workshop on Broadband Advanced Sensor Networks, Boston, MA, Oct. 3, 2005. http://research.microsoft.com/en-us/um/people/zhao/pubs/semantic_services.pdf, last accessed on February 17, 2009.

8. McMullen, D., T. Reichherzer. The Common Instrument Middleware Architecture (CIMA): Instrument Ontology & Applications. In: Proceedings of the 2nd Workshop on Formal Ontologies Meets Industry, Trento, Italy, 2006, pp. 655–663.
9. Preece, A. et al. An Ontology-Based Approach to Sensor-Mission Assignment. http://www.csd.abdn.ac.uk/~sleeman/published-papers/publications-2007/p176-ITA-ADP.pdf, last accessed on February 17, 2009.
10. Russomanno, D., C. Kothari, O. Thomas. Building a Sensor Ontology: A Practical Approach Leveraging ISO and OGC Models. In: Proceedings of the 2005 International Conference on Artificial Intelligence, CSREA Press, 2005, pp. 637–643.

Memetic Algorithms

Todor Tsonkov

3rd year student at Computer Science, Sofia University
ttsonkov@gmail.com

Abstract. The main topic of the paper is a new type of heuristic algorithms – memetic algorithms. The similarity between memetic algorithms and genetic algorithms has been seen in details. The main goal of the paper is to show the advantages of memetic algorithms over similar others and a detail description of the body of a memetic algorithm. There's made a comparison between memetic and genetic algorithms when solving different problems. The future development of the paper will include solving more problems and a detailed investigation of the individual learning procedure.

Keywords: Memetic, Heuristic, Evolutionary, Genetic, Travelling Salesman.

1 Introduction

The term meme is introduced for the first time Richard Dawkins' brilliant book “The Selfish Gene”.

Memetics is a science based on evolutionary models for transferring data. The meme is a cultural unit, symbol or practice or other ways that can be imitated. It comes from the Greek word mimema (imitation).

Memetics and genetics have a lot in common. The reason for this is that their basic units – the gene and the meme have the following properties that make them successful replicators:

1. Ability to change themselves (mutation).
2. Ability to create own copies (replication).
3. Ability to have a different survival fate (and it can be evaluated using a fitness function).

Yet, memes have the properties necessary for evolution, and thus meme evolution is not simply analogous to genetic evolution, but a real phenomenon subject to the laws of natural selection. The main (and most important difference) is of the spread – the genes are spread vertically in the generations while the memes can be spread both horizontally and vertically (as long as their fitness function is very high). Therefore the rate of the mutation and the spread in memes is much higher than in genes.

2 Memetic algorithms – definition, basic terms

Memetic algorithms are a population method based on heuristic search in optimization problems.

It's proven that they are a way faster than genetic equivalents in some problems. In their bases they combine local heuristics with operations for recombination. As they are very well fit for multiprocessor systems some scientists consider they the parallel genetic algorithms.

A common pattern for memetic algorithms goes as follows:

```
Procedure Population-Based-Search-Engine
begin
Initialize pop using GenerateInitialPopulation();
repeat
newpop ← GenerateNewPopulation(pop);
pop ← UpdatePopulation(pop,newpop);
if pop has converged then
pop ←RestartPopulation(pop);
endif
until TerminationCriterion();
end
```

An alternative version of the pattern is:

```
Procedure Memetic Algorithm
Initialize: Generate an initial population;
while Stopping conditions are not satisfied do
    Evaluate all individuals in the population.
    Evolve a new population using stochastic search operators.
    Select the subset of individuals, $\Omega_{il}$, that should undergo \the individual improvement procedure.
    for each individual in $\neg\Omega_{il}$ do
        Perform individual learning using meme(s) with frequency or probability of $f_{il}$, for a period of $t_{il}$.
        Proceed with Lamarckian or Baldwinian learning.
    end for
end while
```

The difference between the standard genetic algorithms and memetic is the individual procedure for improvement. In the paper we will see it in details and why it can be a huge improvement.

Let's see some details about creating memetical algorithms – chromosome representation, crossover, mutation.

Chromosome representation. The chromosomes should contain information about the solution they represent. The most common way of encoding is the binary string. The chromosomes look like this:

Chromosome 1	1101100100110110
Chromosome 2	1101111000011110

Each chromosome has a binary string. Each bit in the string might be a characteristic of the solution. Another way is that the string is a number.

Of course, there are many other ways for encoding. It depends mainly on the problem we need to solve. For instance, we can encode an integer or a real number or some kind of permutations.

Crossover. After we decide what kind of encoding we will use, we can make a crossover. During crossover we choose genes from the parent chromosomes and create a new generation. The simplest way to do this is by choosing a random point and the first part is from the first parent, the next from the second.

The crossover might look like this (| is the crossover point).

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Generation 1	11011 | 11000011110
Generation 2	11011 | 00100110110

There are other ways to make a crossover, for instance choosing more points for crossover. It can be made much more complicated depending on the representation of the chromosome. The choice of crossover can improve drastically the run time of the genetic algorithm.

Mutation. After the crossover we'll have a look at the mutation. Mutation changes the new generation randomly. In the binary string we can change randomly two bits from 1 to 0 and vice versa. An example follows:

Original Generation 1	110 1 111000011110
Original Generation 2	110110 0 1001101 1 0
Mutated Generation 1	110 0 111000011110
Mutated Generation 2	110110 1 1001101 1 0

Mutation depends on the encoding as much as from crossover. For instance, when we encode permutations mutation can be the change of two genes.

3 Local Search Procedure

The existing, successful methods in approximate iterative optimization fall into two broad classes: local search, and population-based search, and most implementations of Memetic Algorithms include local search methods, it should firstly discuss local search algorithm. A local search algorithm starts from a configuration s0 generated at random or constructed by some other algorithm. Subsequently, it iterates using at each step a transition based on the neighborhood of the current configuration. Transitions leading to preferable configurations are accepted, i.e., the newly generated configuration becomes the current configuration in the next step. Otherwise, the current configuration is kept. This process is repeated until a certain termination criterion is met. Typical termination criteria are the realization of a pre-defined number of iterations, not having found any improvement in the last m iterations, or even more complex mechanisms based on estimating the probability of being at a local optimum. Due to these characteristics, the approach is normally called "hill climbing". An example goes as follow:

```
Procedure Local-Search-Engine(current)
begin
repeat
new < GenerateNeighbor(current);
if (Fg(new) < Fg(current)) return
current -> new;
endif
until TerminationCriterion();
return current
end
```

We can use different approaches to create a suitable local search procedure. It depends on the conditions we have in our problem. The most important issue we have to consider is to see how much do we need for the individual learning procedure and the time necessary for the genetic algorithm to produce the required results. Therefore the exact amount of time consumed for each of the two parts is very important. In Hart W. E. (1994). *Adaptive Global Optimization with Local Search.*, is shown that after different situations the best way to use the individual learning procedure when its calculation value is not too big. Another heuristic algorithms are: Simplex Method, Newton/Quasi Method, interior point methods, line search.

4 Classical Problems

We will see by solving examples the difference between genetic and memetic algorithms. The first problem is the classical Vertex Cover Problem. We need to find such a subset of vertices in a graph that each edge has at least one of the vertices from this subset and this subset is minimal (in terms of its cardinality). We will see an approach trying to solve it.

The individual learning procedure goes as follows:

Procedure 1:

```
Procedure VertexCover( Graph G )
      set S.
      repeat:
            Select randomly a vertex of maximal degree
            Add it to  S.
            Delete all edges incident to this vertex.
      until:
            Graph is not empty.
      return S.
```

How do we improve this algorithm, e.g. How do we find a solution with less vertices?

The memetic algorithm goes as follows:

Procedure 2:

```
Procedure Vertex-Cover-Memetic( graph  G )
      Initialize initial population by using Procedure1
      repeat:
            new-population =
            GenerateNewPopulation (initialPopulation)
            initalPopulation =
                  TaketheBestFrom(new-population,
                        initialPopulation )
            improvebyHillClimbing( initialPopulation )
      until:
            TerminationCriterion.
```

We test procedure2 on random graphs with 1500 vertices and 4500 edges.

The first time generated Population is: { 934, 932, 936, 944, 936, 933, 933, 940, 937, 940, 938, 933, 936, 941, 942 } (number of vertices). After the memetic algorithm we get 902 vertices.

In another distribution with 1500 vertices and 3000 edges we have a following distribution of vertices { 818, 819, 820, 814, 819, 821, 815, 824, 813, 820, 812, 822, 817, 817, 822} and after the memetic algorithm we have 790 vertices.

Another classic problem is the Travelling Salesman problem. After testing with different memetic algorithms we have an improvement by about 5% in terms of distance.

5 Conclusion

In the paper, the design of memetic and the individual learning procedure have been seen in details. The future of the paper will include solving more NP-complete problems and trying to make some classes of problems ordered by the most effective learning procedure.

References

1. Ozcan, E.; Onbasioglu, E. (2006). "Memetic Algorithms for Parallel Code Optimization". International Journal of Parallel Programming 35 (1): 33-61.
2. Moscato, P. (1989). "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms". Caltech Concurrent Computation Program (report 826).
3. Ichimura, T.; Kuriyama, Y. (1998). "Learning of neural networks with parallel hybrid GA using a royal road function". IEEE International Joint Conference on Neural Networks 2: 1131-1136.

Data Streams Model

Vladimir Dimitrov
Faculty of Mathematics and Informatics, University of Sofia,
5 James Bourchier Blvd., 1164 Sofia, Bulgaria
cht@fmi.uni-sofia.bg

Abstract. Traditional Database Management Systems (DBMS) are suitable for execution of single queries on finite data sets stored in the database. There are many new applications, like network monitoring, financial analysis, manufacturing systems and sensors networks, which have to execute long or continuous queries on infinite and unbounded data streams. These streams could be very rapid and their characteristics and load with queries could vary with time and the resources could be bounded at the same time. This paper investigates data management for data streams class of applications.

Keywords: Data Streams, Data Base Management Systems.

1 Introduction

Data streams are an extension to the traditional DBMS. Internet technology has made communications among computers a routine operation. As result of that new class of applications emerged that could not be supported in the framework of traditional DBMS. Typical database system is data storage. New data in this storage are inserted with a subset of data manipulation language or with special data load utility. This means that new data are inserted only under DBMS control. But in the newly emerged applications DBMS cannot control the rate at which data arrive. Such applications could be a program processing the data from a network of radiation level sensors distributed all over the world.

There are many different kinds of sensors, whose outputs have to be read and reviewed together. Such a system is tsunami warning system that uses many sensors recording ocean level at some part of the second, and uses the seismic sensors from all over the world that records earth shakes. Another example is security cameras networks used in the cities whose video is recorded and analyzed for violations.

Satellites send to the earth huge data streams, which sometimes are petabytes in the day. Scientists do not want to discard such data and store raw data in archive systems. The last one is called in joke “write only storage”. Useful part of these data are retrieved when they arrive and are stored in more easy accessible places or are distributed to scientist, that have put standing queries for identification of different kinds of data.

In data stream applications, data are in form of one or several data streams. Such kinds of applications are:

- **Click streams**. The sources of these streams are user interactions with a portal web site. These interactions could be analyzed for increased interactions with a given hyperlink that could mean that the hyperlink is broken or that there is an increased popularity on it. Search engine can analyze information to which hyperlinks point to evaluate which of them are most attractive.

- **Package streams**. Sources and targets of IP packages crossing some router could be analyzed. Unusual big traffic of packages to a given source could be warning for DoS (Denial of Service) attack. With some preview of history of targets could be predicted state of the net and eventually packages could be rerouted.
- **Sensor data**. There are many kinds of sensors whose output have to be read and analyzed. For example, tsunami warning sensors register ocean level at every part of the second. This network includes seismic sensors all over the world that register earth shakes. In the cities, there are networks of security cameras that send video signals that are analyzed for law violations.
- **Satellite data**. Satellites send every day to the earth huge data streams that frequently are petabytes. The scientists usually do not want to discard these raw data and they are stored in archiving devices. The last one are sometimes called in joke "write only storage". The useful part of the data are extracted via standing queries when data arrive and is stored in more accessible storage or is distributed among scientists.
- **Financial data**. Stoke exchange, shares and other financial instruments are registered as tuple streams where every tuple represents a financial transaction. These streams are analyzed by specialized software searching for events or patterns of dealers' behavior. Successful dealers have access to big data volumes that are rapidly processed. That is important, because opportunities on the market usually are available only for a part of the second.

2 Data Stream Management System

For execution of queries on streams some new mechanisms are needed. In traditional DBMS is impossible to store high rate data and at the same time to execute queries with some relational language as SQL. The problem is that it is not clear exactly what some queries are, for example what is a join of two streams that never end?

Data Stream Management System (DSMS) [1] is an extension to DBMS. Its common structure is presented at Fig. 1.

DSMS accept as input data streams and queries. Queries are of two types:

1. Conventional ad-hoc queries, and
2. Standing queries that are stored in the system and are executed on the streams all the time.

Independent of what a kind are the queries, they have to be expressed to answer on limited part of the streams. DSMS could not store all the data and search in the streams for any given time interval, instead it store a sliding window on every input stream. These sliding windows are stored on the secondary storage of the working storage. Such a sliding window could be the data from the last 24 hours from a given sensor. Oldest data could be discarded, averaged (for example with the mean day data value) or copied to the permanent storage (archived).

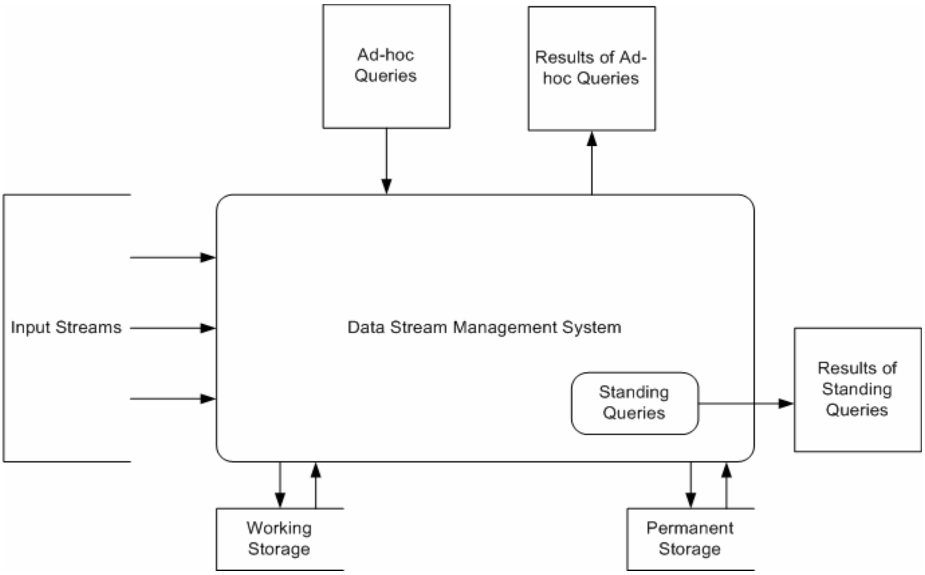

Ad-hoc Queries
Results of Ad-hoc Queries
Input Streams
Data Stream Management System
Standing Queries
Results of Standing Queries
Working Storage
Permanent Storage

Fig. 3. Data Stream Management System.

An ad-hoc query could asks for the mean radiation levels from all North Korea locations, and this query could be answered since all the data from all the streams from the last 24 hours are available in the working storage.

A standing query could warn if some data in the stream get out of some bounds. This could be done, because all the data from every stream enter the system, they could be compared with some boundaries and if they are out some warning could be generated. Such a query could be executed by the streams themselves, but it is possible a search in the working storage to be done, for example if we want to be warned when the mean value of some stream from the last 5 minutes get out of boundaries.

DSMS is an extension to DBMS. The challenges to it are:

- Data streams are structured in records data as conventional relations, but it is not possible to apply standard relational semantics to these complex unlimited queries on these data. It is needed a new semantic and a query language on streams and relations.
- Declarative queries have to be translated in physical plans, which are flexible enough to support fine optimizations of fine granulated execution plans. The last ones consist of operations, queues and synopsis.
- High performance requires DSMS to have facilities for sharing states and calculations inside and among the plans. Constraints on the streams (ordering, clusterization, referential constraints) could be derived and used to decrease resource usage. For that purpose corresponding techniques have to be developed.
- Data, system characteristics and query load might change with time and it is needed adaptive approach to achieve good execution. For that purpose, a subsystem for permanent monitoring and re-optimization is needed.

- When the speed of input data exceeds DSMS capacity to return exact results to the active queries, the system has to apply load scheduling using approximations that decrease the precision in acceptable boundaries. Strategies for such approximations have to be developed.
- The standing queries are naturally long running and DSMS needs of tools for monitoring and manipulation of the queries during its execution. These tools have to be used not only by administrators, but the users must have access to them too.

3 Data Stream Model

The data stream model is useful for discussion of algorithms on data streams. About the streams we suppose that:

- Every stream is a sequence of tuples. The last ones have fixed relational schema (list of attributes) as in relations. The difference is that relations are finite, but the streams could be unbounded.
- Every tuple is associated with arrival time, i.e. the time when it starts to be available for processing in DSMS. The last one can put it in the working storage, in permanent storage or to discard it. The tuple could be processed in very simple way before its storage.

On every stream could be defined a sliding window (simply window) that is a set of last arrived tuples. This windows could be time-based with a constant `t`, which means that it contains tuples arrived in the interval from the current time till current time minus `t`. Or the window could be tuple based on some integer `N` for the last `N` arriving tuples. These windows could be denoted with `S[w]`, where `w` is window descriptor one of the following:

- `Rows n`, that means last `n` tuples of the stream, or
- `Range t`, tuples arrived in the last time `t`.

For example let `Sensors(sensId, temp, time)` is a stream of tuples containing the temperature `temp` read from a sensor identified with `sensId` at the moment `time`. It is more realistic every sensor to have its own stream, it is also possible all the streams to be joined in one outside of DSMS. The expression `Sensors[Rows 1000]` is a window on the stream `Sensors` for the last 1000 tuples. The expression `Sensors[Range 10 Seconds]` specifies a windows on the same stream for the tuples arrived in the last 10 seconds.

4 Converting Streams to Relations

Windows permit us to convert streams into relations. More precisely, windows expressions are describing relation at every one moment. The content of this relation is changing very fast. For example let see the window `Sensors[Row 1000]`, at every moment when new tuple is arriving from `Sensors`, that tuple is inserted and the oldest tuple is deleted. For the expression `Sensors[Range 10 Seconds]`, the tuples are inserted when they arrive and are deleted 10 seconds after their arrival.

Windows expression could be used in one extended SQL for streams. In the next example such an extension gives an idea about such extended SQL.

For example, lets we want to know what the highest temperature is recorded and registered by DSMS in the last one hour. We have to specify suitable time-based window and to formulate query on it as it is a normal relation. The query looks like:

```
SELECT sensId, MAX(temp)
FROM Sensors[Range 1 Hour]
GROUP BY sensed;
```

This query could be ad-hoc on the window, which has to exist at the moment when the query is issued. The window has to be created at least one hour before the query is issued and forwarded for processing to query processor of DSMS. In other words, to answer the query, the DSMS has to have been collected enough data. DSMS could answer this query at every time if it removes the tuples for which there is new higher temperature for the same sensor.

This query could be a standing one, in which case the resultant relation could be a materialized view that is changing all the time. We will investigate later again how the result of this standing query could be represented.

Windows relations could be combined with other window relations or with other usual relations – such that are not on streams. The next example shows such a combination.

For example, let our DSMS has an input stream Sensors, but in the working space is stored usual relation `Calibrate(sensId, mult, add)` that represent a multiplication factor and addition factor for correcting reads from the sensors. The query:

```
SELECT MAX(mult * temp + add)
FROM Sensors[Range 1 Hour], Calibrate
WHERE Sensors.sensId = Calibrate.sensId;
```

finds the highest calibrated temperature that has been read from any sensor in the last an hour. In this query window relation is joined with the usual relation `Calibrate`.

In just a same way we could join windows relations. The next example illustrates autojoin via subquery, but all SQL features for join expression could be used.

Lets now we want to see for every sensor its maximal temperature in the last an hour and the tuples to contain information about the last moment in which this maximal temperature is recorded. The query is:

```
SELECT s.sensId, s.temp, s.time
FROM Sensors[Range 1 Hour] s
WHERE NOT EXISTS (
    SELECT *
    FROM Sensors[Range 1 Hour]
    WHERE sensId = s.sensId AND
          (temp > s.temp OR (temp = s.temp AND
                             time > s.time)));
```

The subquery checks is there any tuple in the window relation `Sensors[Range 1 Hour]` for the same sensor that is the tuple s and does it has highest temperature or it is with the same temperature, but with a later timing. If there is no such a tuple, tuples like s are added to the result.

5 Converting Relations to Streams

When standing queries are issued, the resulting relations are changed very frequently. Support of these relations as materialized views could lead to many efforts for insertions and deletions, for which no one looks. The alternative is to convert the relation that is result of the query, back in stream that could be processed as any other stream. For example, we can issue ad-hoc queries to create the result only when we need of its value. If `R` is a relation, we define `Istream(R)` as a stream of all tuples inserted into `R`. One tuple appears in the stream at the moment of its insertion. In the same way, we define `Dstream(R)` as a stream of tuples deleted from `R`. One appears in this stream at the moment of its deletion. Tuple update could be represented as one insertion and one deletion at just a same time.

For example, let `R` is a relation constructed with the query from the last example, i.e. it's the relation containing for every sensor its maximal temperature that is registered in the tuples inserted in the last one hour, and the tuple contains the last time when this temperature has been read. Then `Istream(R)` has a tuple for every event when a new tuple has been added to `R`. Note that there are two events that add tuples to `R`:

1. A new tuple arrives for `Sensors` with temperature that at least as high as any tuple in `R` with the same sensor identifier. This tuple is inserted into `R` and at the same time is an element of `Istream(R)`.
2. The current maximal temperature for the sensor `i` is recorded before one hour and there is at least one tuple for the sensor `i` in the stream `Sensors` during the last hour. In this case, the new tuple for `R` and for `Istream(R)` is the tuple of `Sensors` for the sensor `i`, that has arrived during the last hour, if there is no tuple for `i` arrived in the last hour that has:
   a. higher temperature,
   b. the same temperature, but arrived latter.

These two events could generate tuples for the stream `Dstream(R)`. In 1., if there is another tuple in `R` for the same sensor, then this tuple is deleted from `R` and become an element of `Dstream(R)`. In 2., an hour older tuple of `R` is deleted and become an element of `Dstream(R)`.

If we compute `Istream` and `Dstream` for relation like that one constructed with the query from the example, there is no need to support this relation as materialized view. Instead, we could execute queries to its `Istream` and `Dstream` when we need to receive answers.

As an example let's construct `Istream I` and `Dstream D` for the relation `R` from above example. When we need, we can run queries on these streams. For example, let's want to see the maximal temperature recorded for sensor 100 in the last hour. This is the temperature in the tuple `i` for sensor 100, that:

- has time from the last hour
- is not removed from `D`, that is not in `D` bounded from the last hour.

This query could be written as:

```
(SELECT * FROM I [Range 1 Hour]
 WHERE sensID = 100 AND time >= [Now - 1 Hour])
```

```
EXCEPT
(SELECT * FROM D[Range 1 Hour] WHERE sensed = 100);
```

In the query keyword Now means the current time. We have to check is the tuple has arrived in the last hour and the tuple is registered in the last hour. These conditions are not the same, it is possible for a tuple to have been arrived in the last hour, but his temperature to have became maximal for the sensor 100 thirty minutes ago. The reason for that is that higher temperature of t for sensor 100 has been registered 90 minutes ago. Only 30 minutes ago t became highest temperature for sensor 100 in the last 60 minutes.

6 Conclusion

The problematic of data streams has been developed at Stanford University [2]. Then the model has been implemented in the commercial database systems, like Oracle and DB2. For research and investigations the source code supplied by Stanford University is more suitable, because there are many problems that are still not solved.

This paper is an overview of data stream model as presented at [2] and it is starting point for future research. Important research directions are:

- **Distributed stream processing**. In the centralized model, data are moved to the center. The question is "What will happen if the processing is moved to the source?"
- **Recovery**. The problem is that recovery models of DBMS are transactions based, but this model is inapplicable to the DSMS. In DBMS data are not changed when the system does not work, but in DSMS the situation is different, even more, some applications require data not to be lost during system downtime. In DBMS, transactions that are executed at the failure time could be forgotten, but this is not the situation with the DSMS and continuous queries.
- **Approximation improvement**. The main idea is user to set the needed precision.
- **Connection with publication-subscription systems**. Publications could be regarded as streams and subscriptions as continuous queries. The problem here is that DSMS are designed for small number of continuous queries and in the case of publication-subscription systems there are millions of such queries.

Acknowledgments. This research is supported by Project VU-MI-204/2006 "Intelligent Sensor Systems for Security Enhancement" funded by the Bulgarian National Science Fund.

References

1. Babu, Sh. & W. Jennifer (2001) Continuous Queries over Data Streams. SIGMOD Record, 30 (3).
2. Stanford Stream Data Manager, http://infolab.stanford.edu/stream

Minimal Set of Metadata for Visualization of Multimedia Files

Peter Armyanov
"St. Kliment Ohridski" University of Sofia, Faculty of Mathematics and Informatics
Sofia 1164, Bulgaria
parmyanov@fmi.uni-sofia.bg

Abstract. This paper studies the minimal set of metadata required for describing a multimedia clip in such a way that a virtual copy of the file in a different container format can be easily created. With the purpose of make this process fast and easy the types of container formats are examined with special attention to those ones widely used in video editing industry. Comparison of format characteristics is made and their base intersection is defined together with information for its description.

Keywords: virtualization, multimedia, metadata, real-time processing.

1 Introduction

In the contemporary world multimedia technologies are widely spread. Numerous software companies develop various systems for creating, editing or reproducing of video or audio files. Tending towards the integration of more options in their products and the accomplishment of high performance, most of the companies develop their own formats for storing processed multimedia. This makes the usage of several different programs inconvenient, due to the many conversions of data from one specific file format to the other.

The following paper is dedicated to one of the possible solutions to the problem – the creation of virtual multimedia files, representing video and audio data in a differing format from the one, in which they were initially saved, without them to be actually present on a physical media. A copy of the file may be kept on the physical media in one format, and the others to be obtained virtually, in real time. This strategy benefits from the fact that the contemporary computer workstations have large amount of memory and powerful processors. To accelerate the creation of virtual files, a preliminarily processed metadata from the original file must be stored in order to build the virtual files and make further processing more efficient and easy. This paper makes a selection of the needed set of preliminarily processed metadata for a multimedia file, which enables the creation of physical or virtual copies of a given multimedia file in a different format.

A main requirement for the process of virtualization is to take place in real time, with accordance to the processing of data, without any intervention of the user. This can be achieved by cutting down the task to repacking the data, without changing the manner or algorithm used to compress it. This operation is not always possible, because the various formats for storing multimedia information may wrap data, recorded by using different compressions, as well as store different types of information.

2 Types of container formats

Up to this moment the container multimedia formats can be divided into two major categories according to their functions: formats facilitating transport and storage of multimedia information, and formats facilitating the modification (mostly linear editing) of multimedia. The formats may be additionally divided to professional and customers. Furthermore, we may divide the transport formats to streaming and indexed.

The main target of the streaming formats is to enable transfer of continuous stream of information, so that it would be possible for users to begin reading the stream from a random point and to be able to reproduce the multimedia transported in the stream. Because of this requirement the descriptive information in this type of multimedia formats is repeated on a regular basis in the stream. Such formats are most commonly used in the digital television transmission. They do not allow random access of data. Typical examples are the formats of the MPEG family – such as mpeg2 TS which carries the satellite television.

The index storage formats are designed to enable easy access in a randomly relative time in the file. This is usually achieved with an index table with records for each frame of the media. These formats are used most commonly as target formats for conservation and are useful for previewing and editing. Because the descriptive information is stored most commonly in the beginning or in the end of the file, these files can difficultly be reproduced if a part of them is not accessible and cannot be used for stream transport. Often, these formats are designed to allow storage of additional text information, like subtitles for example. Typical formats of this category are AVI, VOB, as well as the professional video processing format MXF.

The formats facilitating the editing are most often designed especially for a given professional system for video or audio editing and contain numerous system specific metadata. The most important among them is the information for the relative time of recording each frame, called timecode. These formats enable an easy split of media at a random point. They can also contain many different media tracks of one single type, for example several sections with video information with a different compression rate or format. Many of the formats in this category can be supplemented to the index transfer formats. The most common formats of this type include: MXF, Apple MOV and its open version - MP4.

A comparison of some popular file formats based on the type of information which they can store and the additional options which they provide is shown on Table 1.

ontainer format	Supported video compression types	Supported audio compression types	VBR video supported	VBR audio supported	Inplace editing	User defined information	Subtitles	Streaming	Random access
P	MPEG-4, H.263, H.264	AMR, AAC	yes	yes	no	no	no	yes	no
F	no limitations	problems with Vorbis	yes	yes	no	no	yes	yes	limited
I	no limitations	problems with Vorbis	yes	yes	no	no	yes	no	limited
sh Video	Sorenson, VP6	Raw, ADPCM, LPCM, MP3	no	no	no	no	no	yes	no
troska	no limitations	no limitations	yes	yes	no	yes	yes	yes	yes
EG-2 PS	MPEG-1, MPEG-2	MPEG-1, MPEG-2, AC3	yes	yes	no	no	yes	no	yes
EG-2 TS	MPEG-1, MPEG-2, H.264, MPEG-4	MPEG-1, MPEG-2, LPCM, AAC	yes	yes	no	no	yes	yes	no
V	no limitations	no limitations	yes	yes	yes	yes	yes	yes	yes
4	MPEG-1, MPEG-2, H.263, MPEG-4	MPEG-1, MPEG-2, Vorbis, AAC	yes	yes	yes	yes	yes	yes	yes
F	DV, raw, MPEG SD, MPEG HD, XdCAM, IMX	LPCM, AES3, MPEG-1, MPEG-2	no	yes	no	yes	no	no	yes
G	no limitations	no limitations	no	yes	no	yes	yes	yes	yes
B	MPEG-2 part 2	AC-3 LPCM, MPEG-1, MPEG-2	yes	yes	no	no	yes	no	limited

Table 4. Comparison of capabilities of some multimedia formats

According to different capabilities and characteristics of container formats inability to convert given multimedia clip, stored in specific format, in another without loss of information or media transcoding may arise. Usually in the virtualization process media changes are allowed only when the processing could be accomplished without exorbitant delay. The virtualization is based on minimal quantity of additional information describing original media file and changes are allowed only when the original quality of picture and sound is kept.

3 Required metadata.

To make a virtual or physical copy of a multimedia file in a format different from the primary one is needed information describing the file characteristics at all and each of its media tracks. In files, containing video clips, there are several different types of information. The most important ones are the information about the images – video data - and the information about the sound – audio data. Together with this data in the multimedia files often persist additional descriptive information – metadata. This information describes specific parameters of medias stored in the file and of the clip as a whole (for example clip name, date and place the clip is shot, program or device the clip is created with, shooting angle and so on). Another type of metadata is user-metadata – information used by the clip creator for specific purposes. In most of the formats the user metadata is not formally specified and the programs that read and store this information use their own data representation, ignoring all other representations. That's why, retaining of this information in process of virtualization is not possible in most of the cases.

In the file formats, used by motion picture and television industry often is stored information about relative time of each frame in the original material – timecode. This information is very useful in the process of editing the material to the form suitable for broadcasting or previewing.

The video data can be examined as a sequence of separated pictures – frames. The number of frames stored and displayed for a second is called frame rate. Depending on constancy of the frame rate in the clip the clips can be divided in two categories: Clips with constant frame rate and clips in which the frame rate changes. Formats with variable frame rate are commonly used for low-quality clips, mostly intended for streaming over Internet. Formats used in motion picture and television industry always use constant and predefined frame rate – such as 25 frames per second (fps) for PAL, used in Europe; 29.97 fps (which is exactly 30000 frames for 1001 seconds) for NTSC, used in Asia and North America; 24 fps for motion pictures.

The base audio unit is called sample. This is numerical representation of sound amplitude in a given moment. To represent sound large amount of such units must be stored for a second. In most of the cases, fixed values are used - usually 32000, 44100 or 48000 samples per second. 32000 is the standard for old and unprofessional miniDV cassettes, 44100 for audio discs, 48000 for DVD disks and new DV cassettes [3]. In many audio compression algorithms some given number of sequential audio samples is grouped in a unit called frame, which is treated as atomic unit in audio compression and decompression process.

Two types of timecode are commonly used – the elder one is based on relative position of the material on the tape it is recorded to. This timecode type is called timecode-counter and it maps every frame to a number, representing the tape position at the beginning of the frame record. This approach is not very accurate and is not suitable when digital camera is used for recording. That's why the Society of Motion Picture and Television Engineers (SMPTE) issues the standard SMPTE-12m [7], where a new timecode format is described. This timecode is based on time passed from beginning of the recording process to the moment the current frame is shot. This time consist of hour, minute, second and number of frame in the second. The count of frames shot for a second is constant for the clip and is an integer. When the clip frame rate is not an integer, for example 29.79, special moments are defined in the timecode, when two frames, continuous in time has discontinuous timecode values. For example at frame rate 30000/1001 at every minute two timecode values are skipped, except when timecode is rounded to ten minutes (for example following timecode 0:03:59;29 is 0:04:00;02, but after 0:09:59;29 is 0:10:00;00). Such kind of timecode is called Drop Frame Timecode. According to the standard one timecode record is stored in 32 bits, some of which store time information and some specific description flags. An extension of this format is defined also, which supplement another 32 bits of information for each frame, which can be used for specific customer purposes.

In contemporary formats any other kind of metadata may be included in files. Often, additional information is placed about the author of the video clip, the technical means (hardware and software) with which it was created, the date and time of creation, the project of which the file may be part of, subtitles or text comments about the file, metadata about the source where the used codec may be found or sometimes the codec itself and many others. Some formats give ability to store and represent other types of media – for example, the growing-popularity format of Apple

Corporation QuickTime may keep metadata for 3D objects, as well as for projections on three dimensional screens (sphere or cube). It also includes information about the so-called hot-spots in the clip, by means of which it may be interactively change the way of media visual representation (speed of play, brightness, contrast, gamma corrections, sound volume and others), as well as achieving effects, similar to those of the widely-spread Internet format Active Flash Script [5]. A major part of the enabled features are specific for the given container format, and are not supported by most of the other formats and therefore do not require any virtualization metadata.

The basic characteristics of the video data, which are compulsory and are present in each different representation of video information in a given clip, are:

- Number of frames – length of clip;
- Height and width of the image (frame) in pixels;
- Type of colour representation of digital information. In the video clips often are used additive colour representation – most often RGB or YCrCb. The representations may be linear or logarithmic [3, 4];
- Number of colours a pixel represents – this characteristic is determined by the size of memory in bits, which a pixel uses. This characteristic is often called *depth of colour* or just *depth*. Popular depths for RGB representations are 1, 2, 4, 8, 16, 24, 30 or 32 bits per pixel. For YCrCb there is 8, 10, 12 or 16 bit channels, and each channel may have its own density [4]. The information for a pixel is not always byte aligned and furthermore it may not always be an equal power of two. This imposes the usage of different approaches for pixel distribution in the memory – packed or unpacked. The conversion of one of them to another is a physical reordering of information, without its change – lossless transformation;
- Type of image fields– interlaced or progressive [3];
- Frames per second – frame rate. Most frequently it is 24, 25 or 29.97 (30000/1001);
- Data format – compression type such as DV or MPEG.

Each of these characteristics may be stored directly as a part of the clip description or to be retrieved from the video data, but this requires specific knowledge of data representation and the compression algorithms used.

- The corresponding characteristics for the audio data, which are compulsory, are:
- The number of samples – the length of the clip;
- Sample size in bits – the most frequently used are 8, 16, 24 and 32, as well as with less bits - 4 or 6 and with more bits – 48. Few sound systems are capable of producing sound, represented with more than 24 bits per sample, which is why audio with higher quality is rarely used;
- Number of channels (1 - mono, 2 – stereo etc.) – the sound is divided to different parts called *tracks*, each of them representing one sound channel and having its own description. In some cases stereo sound may be represented as two mono tracks. When an audio track contains more than one audio channel, the track is called *interleaved.* Usually the interleaved tracks are most suitable for ready to be released files, while when editing it's more convenient each track to have its own and independent audio channel;
- Frequency of sound – number of samples per second;

- Type of sample representation – whether a signed or unsigned type is used or a type with floating or fixed point while recording of uncompressed data. Type of the compression, if used.
- Size of a frame in samples and bits, if data is compressed.

The information describing the timecode in a clip must include at least the number of records and the type of format in which information is recorded. Most frequently the number of samples is equal to the number of video frames (each of them has a timecode), but sometimes a timecode based on the sound frames may be used, for example for editing clips composed of audio tracks only.

The covered parameters of a multimedia file represent the minimal set of metadata describing the main types of data – audio, video and timecode. In order to be sufficient for a virtual conversion this metadata must be complemented with data concerning the physical placement of media in the original file – where the video and audio frames are placed, what is their size, what is their organization at physical layer. This information is necessary in order to find the reqested data in the file when it have to be read or changed.

The data is usually divided into blocks, composed of a set of atomic units - frames for the video media and samples for audio. This set may be constant or inconstant in size. Often, the different blocks are not with an equal number of units, but a successive interchange of a given number of successive units may be noticed [2]. In these cases, if a succession of a given number of blocks is taken out, there is a constant number of units in it. Taking into consideration these specifications of multimedia data it may be stated that for the description information about the placement of data (the offset from the beginning of the file) and the size of the block will also be needed, as well as the number of units in it. To optimize the process of editing, a *group* may be introduced, which represents the multitude of a given number of blocks containing if possible a constant number of video frames and audio samples.

The information needed for describing the blocks of data, depending on the number of units and their size is given in Table 2:

Block size / ***Number of elements in block***	**Fixed**	**Variable**
constant	block start offset	block start offset; block size

Variable	block start offset;
	number of elements
		block start offset;
		block size;
		number of elements

Table 2. The information needed for describing the blocks of data

To determine the way this information is stored the following specifications are taken into consideration:

- The number of units in a block may be calculated, if for each block the index of the first element is tracked. Thus the number of the units in all blocks (without the last one) would be the index of the first unit in the next block minus the index of the first unit in the current block. To define the number of units in the last block we may use information about the whole number of units in the track.

- Two different blocks are not often consecutive in the file, which is why it’s not possible to define the size of a given block on its position in the file, on the contrary when we are defining the number of units.
- The position of a given element in the block with a varying size cannot be defined.

For some types of multimedia tracks additional specific information is needed. For example for describing video media, compressed by using temporal dependency, the type of frame must be defined (I, P or B), as well as a description of the offset in the order of showing the frames in accordance to the order of storing them (frame reordering). This information may be defined as general scheme for the clip, or may be placed in each frame [4, 8, 9].

For describing the timecode data is chosen the so-called description of blocks with continuous timecode. These are blocks, which have succeeding timecode entries for each two consecutive units. Most often, when the clip is recorded all the frames have sequential timecodes, but when editing the clip it may happen that two consequent frames have recordings of inconsequent values as a result of inserting or cropping parts of the clip. In this situation the information about the timecode will consist of entries for each block – timecode for the first element and length of the block – number of frames with consecutive entries.

It is also necessary to define the information for description of data in one single track. From the previous report, it is clear that the main descriptive data are:

- Type of the track (audio, video, timecode or other).
- Number of units in track.
- Number of units in a single block if it is constant in information, or a signature that it is variable.
- Number of blocks in the whole track.
- Number of blocks in a single group.
- Number of units in a single group – as it was stated, it is possible that the number of units is variable, but when carefully grouping the blocks, a constant value of number of units may be achieved. Such grouping will accelerate the process of searching of information for an unit by a given number.
- Which part of the track is visible and which – invisible during playback.
- Information about whether the media is in the same file and if not - links to other files or resources.

4 Conclusions

With the help of metadata which may be either present in the multimedia file format, or may be extracted easily when being aware of its specifications, it is possible to built in real-time virtual files in each format, compatible with the initial, which was the target of this paper. Thus, the minimal set of metadata for visualization of multimedia files is defined completely.

References

1. Gilmer B. – „File Interchange Handbook for images, audio and metadata” – Focal Press, 2004
2. Devlin B, Wilkinson J, Welles N – „The MXF book” – Focal Press, 2006
3. Jack K. – „Video Demystified – a handbook for the digital engineer” – Elsevier, 2005
4. Symes P.- „Digital Video Compression” – McGraw Hill, 2004
5. QuickTime File Format Specification v.6 – Apple inc., 2007
6. AVI File Format – Microsoft, 2005
7. SMPTE 12M-1995, Television, Audio and Film – Time and Control Code
8. SMPTE 377M – 2003, Television - Material Exchange Format (MXF) File Format Specification (Standard)
9. ISO/IEC 11172 – Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s: Part 1 – Systems; Part 2 – Video

Acknowledgments.

The paper is supported by Grant 173/2009 from Sofia University Research Fund.

Data Mining in Data Streams

Vladimir Dimitrov
Faculty of Mathematics and Informatics, University of Sofia,
5 James Bourchier Blvd., 1164 Sofia, Bulgaria
cht@fmi.uni-sofia.bg

Abstract. When data streams are processed, there are problems difficult to be solved, even when they are easy solvable for the relations. Here, in more details are investigated windows content presentation – not only the current set of tuples in the window is counted. It is not possible to answer all queries to the window, but if it is known in advance what a kind of queries is supposed to be supported, then the window contents can be compressed and the answers to be in some fixed precision boundaries. There are two major problems of this kind. The first one is binary streams (streams of zeros and ones) and queries that count the number of ones in the window for various time intervals. It is clear that, if the full sequence of bits with their time stamps is stored, then it is possible such queries to be answered exactly. It is possible, this data to be compressed and the answers to these queries to be returned with fixed precision. The counting of the different values could be done in the sliding window. Then there is another group of queries that could not be precisely answered if not all the data are stored in the window. It is possible to have good approximation with the usage of much less space than is the window size.

Keywords: Data Streams, Data Base Management Systems, Data Mining.

1 Introduction

Let's consider a window of one billion integers. This window could be stored in big main memory of four gigabytes and in any way could be stored on the disk without problems. If the queries are only on the last tuples of the stream, one billion tuples are enough, but what we have to do if there are one million of such streams?

For example, we can try to integrate the data from one million sensors located in the town, or if we have a stream of shopping carts and we try to calculate the frequencies in the different time intervals for all sets of items contained in these carts. In this case is needed a window for every set with bit flags denoting is the set contained in the cart or not.

In this situation, the space needed for storing all these windows will overcome all available disk space. Even more, to answer effectively, we have to store all these windows in the main memory. In the last case, only several windows of billion integers or several thousand windows with million integers will exhaust even big main memory. That is why we have to find a way to compress the data in these windows. If the compression is not suitable, we will not answer even simple queries.

For example, let's have a sliding window, in which are stored elements of stream of integers and let's have a standing query on it that warns when the sum of integers in the window exceeds some boundaries. In this case, it is enough to store only the sum of integers in the window to answer the query. When a new integer arrives, we add it to the sum.

The numbers leaving the window have to be subtracted from the sum. If the window is tuple based, then we have to subtract from the sum the last integer, when a new number arrives. If the window is time based, then when the time of a number exceeds, we have to subtract it from the sum.

Unfortunately, if we don't know exactly what integers are in the window or we don't know the order of their arrival (for tuple based windows) or their arrival time (for time based windows), then we could not calculate the sum.

There is a situation from which is clear why we could not compress. If we apply some kind of compression, then two different contents of windows w1 and w2 could have just a same value. Because w1 is different from w2, then there is a moment t, in which the numbers in w1 and w2 are different. Let's see what happens when a new integer arrives in the time t and the time of oldest integer expires. In that moment, we have to subtract from the sum to support sums of w1 and w2. Compressed contents of w1 and w2 do not tell us what exact the contents of w1 and w2 are and it is impossible to support the sums in these cases.

From above mentioned reasons, it is clear that it is impossible to compress the sum of sliding window, if we need to recalculate all the time exact answers for the sum. That means that we have to try compress and answer approximated sum. There are many possibilities to do that, but let's consider the simplest one. Stream elements we can group by 100. For example, the first 100 arrived elements are grouped in one group, then next 100 in the next group and so on. Every group is represented by the sum of its elements. In such a way, compression factor is 100, i.e. the window is represented only with one hundredth of all the integers in the window. Let's assume that the window is tuple based and the number of tuples is multiple of 100, then when the number of elements arrived in the stream when become multiple of 100, then we would have exact sum of the elements in the window – simply we have to sum sums of the groups. When a new number arrives after that moment, a new group is started that is represented by its sum. After that we have only approximated sum of the numbers in the window. That is why in the last group of the window we have only 99 of 100 numbers and we don't know value of the integer that is not yet in the window. An approximation for the removed integer is 1% of the sum of the last group. In such a way approximated sum of all numbers in the window, we calculate multiplying with 0.99 the sum of the last group and summing the sums of all other groups.

When new 49 integers arrive, then only 50% of the last sum would be used. After new 50 integers, the new group is finished and the last one is leaving the window. Therefore, the last sum of the last group will be removed and we shell prepare to start a new group when the next number will arrive. Intuitively, this method seems that works. If the numbers are non negative and there are no big variations in the values of the integers, then assumption that leaving numbers from the group are percent of the group is good approximation. Unfortunately, if the variations are very big or integers are not only positive, then there is no limits how bad approximation of the sum could be.

Let's see what will happen if the integers vary from minus infinity to plus infinity and the last group consists of fifty big negative integers followed by fifty big positive numbers, so that sum of the group is 0. Then approximation of the last sum is only half of the 0 when the negative numbers have left the window, but in fact it is very big and even it is possible to be bigger than the sum of all other integers that follow in the stream.

This approach to the compression can be changed in many ways. For example, we could extend the size of the groups to decrease space needed for their representation. In this way, we decrease precision of the approximation. There are others approaches, with which we can limit approximation errors and to achieve significant compression if there is top boundary for the positive integers of the stream.

2 Bits Counting

Let's consider the next problem: N is the length of the sliding window on the stream of zero and ones. The stream has started at some time and arriving bits are associated with a time positioning them in the stream, for example the first bit is associated with time 1, the next bit with time 2 and so on.

Queries that we can execute are of kind "How many ones there are in the last k bits?" where k is an integer between 1 and N. It is clear that if we store the window without compression, we can answer such queries summing the last k bits. It is possible k to be very big and then the answer will be returned after a long calculation. If with the bits we store sums of some groups of consecutive bits – for example, groups long 2, 4, 8 etc. bits, then we can decrease the time for answering to $o(\log_2 N)$.

It is clear that if we store sums of these groups, then we need of more space than simply storing window elements. An attractive alternative is to store information for the stream in $\log_2 N$ length and to answer queries of the above mentioned type with some acceptable precision. Formally, for $\varepsilon>0$ we can achieve approximation in the interval $1-\varepsilon$ to $1+\varepsilon$ near to the real result. This method will be presented for $\varepsilon=1/2$, but can be generalized for every $\varepsilon>0$.

To describe algorithm for approximated counting of ones, we define directory table with size m. The window is divided between this directories, but it is possible some of the zeros not to be included in any directory. Every directory contains exactly m ones. Directories a presented by the pair (m,t) where m is the directory size and t is the time of last arrived one in that directory. Directories dividing the window must satisfy:

1. Directory size must by power of 2.
2. Directory size does not decrease with time.
3. For m=1,2,4,8,… to some directory with the biggest size, there are one or two directories with a given size, but never more than two.
4. Every directory starts somewhere in the current window, but the last (the biggest one) could be partially outside the window.

Under these assumptions one directory could be represented with $o(\log_2 N)$ bits. Something more, no more than $o(\log_2 N)$ directories could be presented. In such a way, window with size N could be presented in space $o(\log_2 N)$, but not with $o(N)$ bits. To see only $o(\log_2 N)$ are needed, we note the following:

- Directory (m,t) could be presented with $o(\log_2 N)$. First of all, the directory size m is never higher that N. Something more, m is power of two, so it is not needed to represent m, it is enough to store $\log_2 m$ – that means $o(\log_2 (\log_2 N))$ bits. We have to represent not only m, but t – the time of last arrived one in the directory. The time t, in principle, could be very big integer, but it is enough to represent it by module N, because t has to be in window with length N. So, $o(\log_2 N)$ bits are enough to represent m and t. Knowing the time of new arrived ones and current time, we can store them in $o(\log_2 N)$ bits.

- We can have no more than $o(\log_2 N)$ directories. The sum of sizes of all directories is no more than N and there are no more than two directories with the same size. If there are more than $2+2\log_2 N$ directories, then the biggest one is with the size 2N. A directory with two time shorter size must be available, so that so big directory is outside of the window.

Let's now try to do approximation answers to the queries that count ones in the last k bits. First we have to find the directory in which has been arrived bit for the last k time units. We know how many ones are in the directory – this is its size. This directory is partly in the interval and partly out of it. We can't say how much of it is in and how much is out, but we accept that half of it is this state.

As an example, let's consider the Fig. 1. In this figure k=N and directories are distributed as follows: two directories with size 1, one with size 2, two directories with 4, two directories with size 8, and one directory with size 16. The answer is 2x1+1x2+2x4+2x8+8=36. The last directory is approximated to half of it.

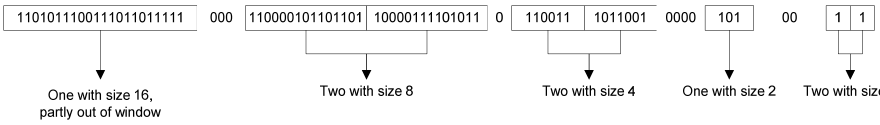


Fig. 5. A window distributed among the directories.

There are two reasons for change when new bits arrive. The first one is easy processed, when new bit arrives and the last directory contains the last arrived bit, which is with N older than the newly arrived bit, and then we remove that directory. The directory cannot participate even partly in the answer of any query.

Let's now check the new bit. If it is 0, then we do not do any changes, but it is possible the last directory to be removed as mentioned above. Let the new bit is 1. We create a new directory with size 1 representing that bit. Now it is possible to have three directories with size 1, which is violation of the rule that we could not have more than two directories of any size. Now we enter in a recursive phase of combining the directories. Let's have the next consecutive directories with size m as follow (m,t1), (m,t2), (m,t3) where t1<t2<t3. We combine the last two directories in one with size 2m. The time of the last arrived bit in the combined directory is the time of the newer directory from the combined directories. This means that (m,t1) and (m,t2) are replaced by the directory (2m,t2). This combination may result in three directories with size 2m, if we yet have two directories with size 2m. In this case, we recursively apply the algorithm for size 2m. The combinations could be spread for no more than $o(\log_2 N)$ time to be done all the combinations. As an example see below:

⇨ 16, 8, 8, 4, 4, 2, 1, 1

⇨ 16, 8, 8, 4, 4, 2, 1, 1, 1

⇨ 16, 8, 8, 4, 4, 2, 2, 2, 1

⇨ 16, 8, 8, 4, 4, 4, 2, 1

⇨ 16, 8, 8, 8, 4, 2, 1

⇨ 16, 16, 8, 4, 2, 1

Let's consider the answer to a query when the oldest directory is totally in query interval. We evaluate its participation in the counting with m/2, which means that error is no more than m/2. The exact answer is the sum of all smaller directories where there is at least one with the size m/2, m/4, m/8, …, 1. This sum is m-1, so the error is no more than (m/2)/(m-1) or near 50%. More precisely 50% is the right top limit.

If we reevaluate the case in which the ones of m are included in the last directory then the error is no more than 1/3. More precisely, the error is (m/2)-1, but not m/2, because there is at least one bit participating in the answer. So, (m/2)-1 is less than m-1, then the error is top limited to 50%.

3 Counting Distinct Elements

Now, we will consider counting of distinct elements in the stream (window of the stream). This problem arises in such applications as:

- Popularity of web sites is counted by number of unique visitor in a month or something like statistics. We can consider visits to a given site like a stream. With a window one month length, we can calculate how many distinct visits take place.
- Let a crawler scan sites. We can accept that the words found in the pages form a stream. If the scanned site is a normal one, then the number of distinct words have to be in some interval, where words are counted not too much or too less. When the site is out of this interval, this means that it is not normal, for example spam site.

To receive the exact answer, we have to store the full window and to apply operator ∂ on it to find out the distinct elements. We need of the count of distinct elements. Even such a simple counting requires the entire window to be stored, but with some methods we can have approximated counting with compression. The next technique counts the distinct elements in the stream not only in finite window. If we want, we can periodically restart the process, for example every month or every time when we start scanning a new site to count the distinct words. The needed tools are an integer N which is big enough to designate all the distinct values in the stream, and hash function h, which maps values in $\log_2$ N bits. We use an integer R which initially is 0. When a new value v arrives from the stream, we do the following:

1. Calculate h(v).
2. Let r is the number of trailing zeros in h(v).
3. If r>R, then R become r.

The number of distinct values is approximately 2^R. Note the following for this approximation:

a. Probability h(v) to have i trailing zeros is 2^{-i}.
b. If there are m distinct elements in the stream, probability R>I is $(1-2^{-i})^m$.
c. If i is much less than $\log_2$ m, then this probability is closer to 1, and if i is much greater than $\log_2$ m, then this probability is closer to 0.
d. In such a way, R will be closer to $\log_2$ m and 2^R our approximation will br closer to m.

These considerations are useful, but they are not right, because the value of 2R is supposed to be infinity or at least to be as big as big is the finite N. Intuitively, for big R when is incremented by 1, probability R to be as big is a half, but value of R is doubled, and every possible value of R donate to the supposed value.

Therefore, we have to overcome the fact that sometimes the value of R is so big that it overestimates m. This problem can be resolved if:

1. We support several approximations of R with different hash functions.
2. Group these approximations in small group and take the average of every group. In such a way, we reevaluate accidentally big R.
3. Take averaged averages of the groups.

Acknowledgments. This work was sponsored by the Ministry of Education and Science of Bulgaria, National Foundation “Science and Research”, contract BY-TH-202/2006: "Multimodal biometric analysis: methods and algorithms”.

References

1. Babu, Sh. & W. Jennifer (2001) Continuous Queries over Data Streams. SIGMOD Record, 30 (3).
2. Stanford Stream Data Manager, http://infolab.stanford.edu/stream

Design Principles of a Digital Library with Learning Materials

Maria M. Nisheva-Pavlova[1], Pavel I. Pavlov[1]

[1] Faculty of Mathematics and Informatics, Sofia University St. Kliment Ohridski,
5, James Bourchier Blvd., 1164 Sofia, Bulgaria
{marian, pavlovp@fmi.uni-sofia.bg}

Abstract. The paper presents some results of an ongoing project directed to the development of a methodology and corresponding software tools for building academic digital libraries. A particular functional model of academic digital library has been discussed. The emphasis falls on some solutions of the large set of problems concerning the development of proper mechanisms for semantics oriented search in multilingual digital libraries. The paper discusses the requirements of the basic types of users of such digital library and suggests some relevant solutions.

Keywords: Digital Library, Electronic Publishing, Metadata, Semantic Annotation, Ontology, Computer Science Model Curricula, Semantic Web.

1 Introduction

Recently computers and networks have changed the ways in which people retrieve information and communicate with each other. In particular, in many areas digital libraries gradually supply the place of traditional ones. At the same time, the developers of digital libraries and tools for access to their content are still faced with a number of challenges. One of these challenges is the execution rate of the user queries. Another challenge is the provision of sufficiently precise and rich in content answers of the user queries. A next considerable challenge is the necessity of development of search methods and techniques which will be appropriate for sets of materials containing documents of different types and multiform content, available in various digital formats.

The paper presents some aspects of an ongoing project which is directed to the development of a methodology and corresponding software tools for building academic digital libraries. The emphasis falls on the functional model of a digital library and the elaboration of search engines appropriate to multilingual digital libraries. The study and the practical experiments are oriented to the development of ProgDL – a digital library with learning materials supporting the courses on Procedural Programming, Object Oriented Programming, Functional Programming and Data Structures and Algorithms designed for Computer Science students at the Faculty of Mathematics and Informatics (FMI) of Sofia University.

The discussed project has the following main objectives:

- To explore the architectural principles of institutional digital libraries and academic digital libraries in particular;
- To study the various aspects of creation of appropriate ontologies oriented to the contents of multilingual digital libraries;
- To define suitable metadata to accompany various types of learning materials in the domain of Computer Programming, taking into account the internationally approved classification schemes and FMI experience;
- To develop a proper framework for application of advanced information technologies and particularly Semantic web technologies in building tools for semantics oriented search in multilingual digital libraries.

The implementation of the project is based on some former results of the authors in the development of software tools for semantics oriented access to digitized collections of manuscripts and digitized archival collections [4–6]. Some actual issues in the areas of personalizing digital library access with preference-based queries [2] and advanced search engine technologies [3] are also taken into account.

2 Functional Model of ProgDL

ProgDL is a typical institutional digital library. It has been under development at FMI in order to provide open access to various kinds of instructional content at BSc level in a wide range of subfields of Computer Programming. The library is intended primarily for Computer Science and Information Systems students at FMI. Its functional structure is shown in Figure 1.

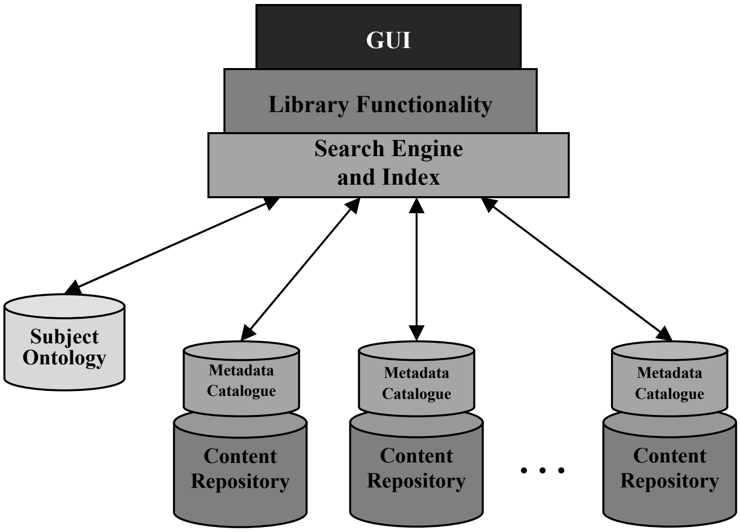


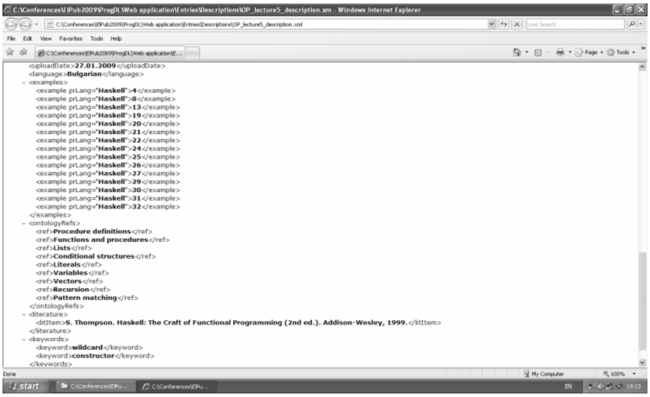
Fig. 1. Functional model of ProgDL

The content repositories include learning materials of different types (textbooks, papers, lecture notes, presentations, exercises, programs, data sets, tests, quizzes etc.) in the fields of data structures and algorithms, procedural, object oriented and functional programming. These library resources are available in various digital formats: pdf, html, plain text, doc, ppt, jpeg etc. Most of them are developed by faculty members, the others are especially selected among the learning materials freely available on the Web. The content repositories are stored in a small number of locations. The learning materials in them are written in Bulgarian or in English language.

The metadata catalogues are destined to facilitate the identification of the needed learning materials by the search engine. They contain descriptive metadata stored in XML format that comply with the IEEE Standard for Learning Object Metadata [7]. Typical examples of relevant attributes of learning materials are: type of the material; style(s) of programming studied in the material; author; title of the material; language(s) (human and/or programming one(s)); digital format; location; version; date of creation; completion status; educational level; principal users for which the material was designed; restrictions on use; semantic annotation – list of concepts from the subject ontology describing the Programming subfields and/or concepts covered or treated by the material. In this way the metadata catalogues support the reusability of all learning materials and facilitate their interoperability.

Each catalogue entry (i.e., each resource description) consists of two equivalent parts in which the element values are texts in Bulgarian or English language respectively. The search engine examines the corresponding parts of the descriptions according to the language of the user query.

Figure 2 shows a selected part of a resource description in XML format.


C:\Conferences\EIPub2009\ProgDL\Web application\Entries\Descriptions\OP_lecture5_description.xm - Windows Internet Explorer
C:\Conferences\EIPub2009\ProgDL\Web application\Entries\Descriptions\OP_lecture5_description.xml
File Edit View Favorites Tools Help
Page Tools
Done
My Computer
100%
start

```
   <uploadDate>27.01.2009</uploadDate>
   <language>Bulgarian</language>
 - <examples>
     <example prLang="Haskell">4</example>
     <example prLang="Haskell">8</example>
     <example prLang="Haskell">13</example>
     <example prLang="Haskell">19</example>
     <example prLang="Haskell">20</example>
     <example prLang="Haskell">21</example>
     <example prLang="Haskell">22</example>
     <example prLang="Haskell">24</example>
     <example prLang="Haskell">25</example>
     <example prLang="Haskell">26</example>
     <example prLang="Haskell">27</example>
     <example prLang="Haskell">29</example>
     <example prLang="Haskell">30</example>
     <example prLang="Haskell">31</example>
     <example prLang="Haskell">32</example>
   </examples>
 - <ontologyRefs>
     <ref>Procedure definitions</ref>
     <ref>Functions and procedures</ref>
     <ref>Lists</ref>
     <ref>Conditional structures</ref>
     <ref>Literals</ref>
     <ref>Variables</ref>
     <ref>Vectors</ref>
     <ref>Recursion</ref>
     <ref>Pattern matching</ref>
   </ontologyRefs>
 - <literature>
     <litItem>S. Thompson. Haskell: The Craft of Functional Programming (2nd ed.). Addison-Wesley, 1999.</litItem>
   </literature>
 - <keywords>
     <keyword>wildcard</keyword>
     <keyword>constructor</keyword>
   </keywords>
```

Fig. 2. Part of a resource description

The elements <ontologyRefs> and <keywords> of the resource descriptions play the role of semantic annotations of the corresponding learning materials. The values of the child elements of <ontologyRefs> are concepts of the subject ontology (names of classes in the subject ontology) which present most precisely the content of the corresponding document.

The concepts of the subject ontology are too general from the point of view of the expectations of the typical users of ProgDL. For that reason one can include in the resource descriptions additional lists of keywords which describe the content of the corresponding documents at the necessary level of abstraction. These keywords are set as values of the child elements of the <keywords> resource description elements.

The next sections describe in brief the rest of the components of ProgDL.

3 Subject Ontology

The subject ontology includes a large set of concepts studied in the University courses in procedural, object oriented and functional programming and data structures and algorithms, with description of their properties and the different kinds of relationships among them. This ontology is based on the Computer Science Curriculum 2008 of ACM and IEEE/CS [1]. Using the curriculum as a guideline, the ontology defines the atomic knowledge units for the mentioned set of programming courses and makes them sharable and reusable. Its current version includes approximately 300 concepts with their relationships. Figure 3 shows a small part of the subject ontology visualized by the OWLViz tab of Protégé/OWL.

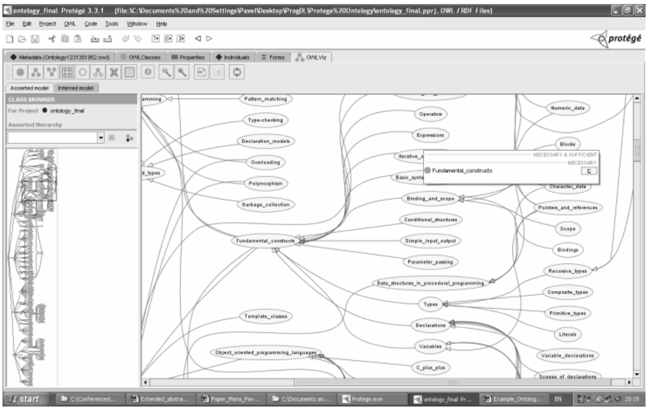


Fig. 3. Part of the subject ontology

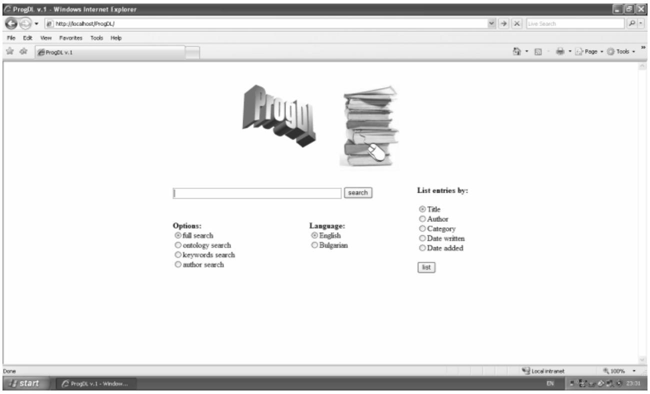
The subject ontology is designed in order to play the role of an information source describing the hierarchy and the other relationships between the main concepts in the discussed domain. A dictionary of synonyms has also been under development with the purpose of providing the search engine with other viewpoints to the conceptual structure of the domain of Programming, Data Structures and Algorithms.

4 User Interface

The library functionality and the user interface of ProgDL are designed in accordance with the expected requirements of the basic types of users of the library. The interface module provides adequate online access to the corresponding library resources (Figure 4).

The current version of the user interface is intended for four types of users:

- FMI students – they may read/download textbooks, open lecture notes and presentations from all public sections of the library as well as all manner of other kinds of materials (lecture notes, presentations, exercises, programs, data sets, quizzes, tests) from fixed public library sections;
- FMI lecturers – in addition to the students' access rights, they may upload materials to fixed public sections as well as create and update private sections and use materials in some of them;
- librarians (library administrators) – they have full access to all public resources of the library (may download and upload materials destined for all public sections of the library);
- general citizen – they may read and download public materials of fixed types (textbooks, open lecture notes and presentations).


ProgDL v.1 - Windows Internet Explorer
http://localhost/ProgDL/
File Edit View Favorites Tools Help
ProgDL v.1
search
list
Done
Local intranet
100%
start

Options:
[✓ full search]
ontology search
keywords search
author search

Language:
[✓ English]
Bulgarian

List entries by:
[✓ Title]
Author
Category
Date written
Date added

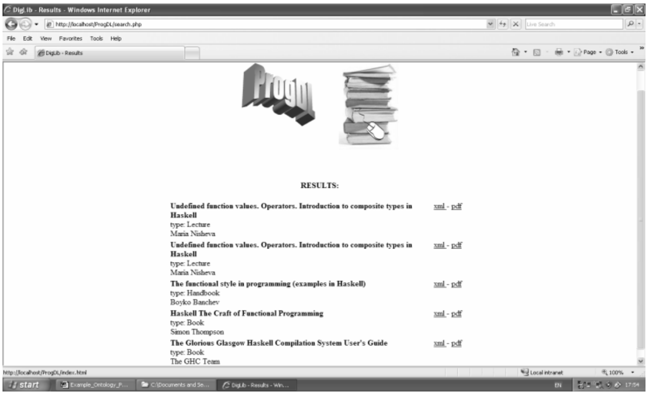
Fig. 4. User interface of ProgDL (students' point of view)

All types of users may choose the language (Bulgarian or English) of their queries to the search engine of ProgDL and then formulate them in the selected language.

5 Working Principles of the Search Engine

The purpose of the search engine is to provide adequate access to the complete palette of resources stored in ProgDL.

The search engine maintains several types of search and document retrieval within ProgDL. The user queries define restrictions on the values of certain metadata attributes of the required learning materials. Generally the search mechanism may be formulated as follows: the document descriptions included in all permissible for the user sections of the library are examined one by one and these descriptions which have a specific element (determined by the type of the user query) with a value matching the user query, are marked in order to form the search result. The matching process is successful if the value of the element or the value of one of its child elements is equal to the user query. The documents pointed by the marked descriptions are retrieved and the user is given an access to these documents and their catalogue descriptions.


DigLib - Results - Windows Internet Explorer
http://localhost/ProgDL/search.php
File Edit View Favorites Tools Help
DigLib - Results
ProgDL
http://localhost/ProgDL/index.html
Local intranet
100%
start

RESULTS:

Undefined function values. Operators. Introduction to composite types in Haskell xml - pdf
type: Lecture
Maria Nisheva

Undefined function values. Operators. Introduction to composite types in Haskell xml - pdf
type: Lecture
Maria Nisheva

The functional style in programming (examples in Haskell) xml - pdf
type: Handbook
Boyko Banchev

Haskell The Craft of Functional Programming xml - pdf
type: Book
Simon Thompson

The Glorious Glasgow Haskell Compilation System User's Guide xml - pdf
type: Book
The GHC Team

Fig. 5. Some search results for the query "composite types"

The current implementation of the search engine supports four types of search and document retrieval:

- full search – search and retrieval of all available learning documents, ordered by title, by author, by category, by date of creation or by date of inserting in the library;
- author search (search and retrieval of the documents created by a given author) – the search is performed in the value of the element <authors>;
- ontology search – the search is performed in the value of the element <ontologyRefs>;
- keywords search – the search is performed in the value of the element <keywords>.

During the ontology search the user query is augmented with regard to the concepts searched out in the semantic annotations of the required learning materials. The more specific concepts from the subject ontology are added to the original one in the resulting query. Then the search engine retrieves all library documents containing in their descriptions at least one component of the augmented query as the value of a child element of <ontologyRefs>. In this way the ontology search enables one to find documents described by ontology concepts which are semantically related to the concept defining the user query.

Fugure 5 shows a screenshot displaying part of the ontology search results for the query "composite types".

6 Conclusion

The most considerable results of the discussed project obtained to the moment may be summarized as follows:

- A functional model of an academic digital library was proposed. This model provides tools for semantics oriented access to learning and research materials in different digital formats;
- A prototype of ProgDL – an academic digital library with learning materials in the areas of data structures and algorithms, procedural, object oriented and functional programming, was developed.

The implementation of the project will help to enhance the exchange of teaching innovation and thus will improve the overall teaching quality in Computer Science at FMI. It will also increase the students' learning experience and their graduation rates.

Acknowledgments. This work has been partly funded by the Sofia University SRF within the "Methods and Tools for Building Academic Digital Libraries" Project, Contract No. 038/2009.

References

1. ACM Education Council, IEEE Computer Society Educational Activities Board. Computer Science Curriculum 2008: An Interim Revision of CS 2001. http://www.acm.org/education/curricula/ComputerScience2008_020309.pdf, last accessed on March 4, 2009.
2. Georgiadis, P., N. Spyratos, V. Christophides, C. Meghini. Personalizing Digital Library Access with Preference-Based Queries. ERCIM News, Vol. 66 (July 2006), pp. 34-35.
3. Lervik, J., S. Brygfjeld. Search Engine Technology Applied in Digital Libraries. ERCIM News, Vol. 66 (July 2006), pp. 18-19.
4. Nisheva-Pavlova, M., P. Pavlov. Tools for Intelligent Search in Collections of Digitized Manuscripts. In: M. Dobreva, J. Engelen (Eds.), "From Author to Reader: Challenges for the Digital Content Chain. Proceedings of the 2005 ELPUB Conference". Peeters Publishing Leuven, 2005, pp. 145-150.
5. Pavlov, P., M. Nisheva-Pavlova. Knowledge-based Search in Collections of Digitized Manuscripts: First Results. Proceedings of the 10th ICCC International Conference on Electronic Publishing (Bansko, 14-16 June 2006), FOI-COMMERCE, Sofia, 2006, pp. 27-35.
6. Nisheva-Pavlova, M., P. Pavlov, N. Markov, M. Nedeva. Digitisation and Access to Archival Collections: A Case Study of the Sofia Municipal Government (1878 – 1879). Proceedings of the 11th International Conference on Electronic Publishing (Vienna, Austria, 13-15 June 2007), ISBN 978-3-85437-292-9, 2007, pp. 277–284.
7. The IEEE Standard for Learning Object Metadata. http://ltsc.ieee.org/wg12/20020612-Final-LOM-Draft.html, last accessed on March 4, 2009.

Analysis of the Need of Data Warehouse Creation

Snezana Savoska[1],
[1] Faculty of Administration and Management of Information systems, University "St.Kliment Ohridski" Bitola, Partizanska bb, 7000 Bitola, Macedonia

Abstract. Data Warehouse is a special tehnological environment that integrates data from intennal transactional company databases, external sources of data in company's repozitorium which is convinient for the data analysis, periodical and ad-hoc reports and decision making. For the strategic decission, making purpose and solving unstructured problems, we use the agregated data from the data warehouse and data marts. Data warehouse becomes the central point of the comapny's demands for information and visual data reprezentation. The concept of Data Warehousing is especially provided with new technology of relational databases with parelel processing.

Keywords: Data Warehouse, Data marts, decision making.

1 Introduction

Data Warehouse is a special technological environment that integrates data in the form convenient for analysis, periodical and ad-hoc reports. Also it makes it easy to process the reporting of managers, analytical staff and decision making process. For the strategic decision making purpose and solving unstructured problems, usually the transactional information systems aren't being used in the company's every day operating systems, but we need to use the aggregated data for the given period of time, collected in the Data Warehouses. The expansion of concept of Data Warehousing is especially provided with the technology of relational databases parallel processing model.
In fact, the Data Warehouse is an adapted reproduction of data of the transactional information systems, especially structured for queries, analysis and reporting. The transactional information data and Data Warehouse data are different entities. The data in the Data Warehouse is updated periodically. Data Warehouses are huge repositories that grow up in enormous range and have a specific structure. Although they come from transactional information systems, are transformed in a specific format suitable for reporting and data visualization. The data is taken from different sources, sometimes from different servers from different operating systems. Data can be loaded in the data warehouse using specific procedures and some transformations in a specific data structure.
They are convenient for periodical, chronological and ad-hoc reports and they are user-friendly for managers and analytical staff for the companies. For this purpose,

we often create additional aggregated tables which are capable of giving prompted information and acceptable data visualization.

The Data Warehouses become the central repositories of all organizational needs for information used for new relationship research, trends and hidden values. They are a common focus point of all organizational members; they enhance business knowledge, very important for the current knowledge era.

The saving data format is different from this transactional information data format. The Data Warehouse is not usually normalized; it can include relational databases, multidimensional databases, flat files, hierarchical databases and object databases. Big part of Data Warehouses is used for post-decision monitoring of effects of decision making called operation analysis. Also, the market pressure, the software vendors and industrial experts make a very strong influence for Data Warehouse and modern data mining tools affirmation (Figure 1). This affirmation is in the direction of development of the company and sectors data warehouses in all forms, using specific data mining tools and enabling data visualization in the decision activities [5].

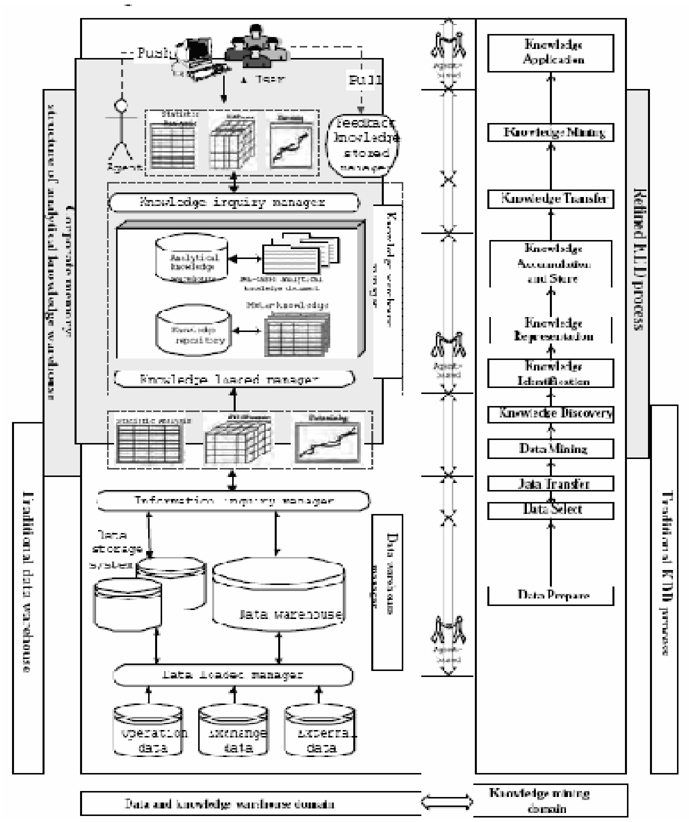


Fig. 1 Data Warehouse as a center of organizational demand [5]

2 The role of Data Warehouse in the company

The Data Warehouse is the target of the data acquisition as a source of a company's data delivery. Wholly looked, Data Warehouses subject it's use, to some primary functions that are in fact its tasks

- The data warehouse directly supports the different business rules in the company and in this way, supports managers and analytical staff in the process of management and decision making. The data warehouse properties must be flexible and easily adaptive to the business rule changes. It includes adding new entities; hierarchical relations dislocate and change the static entities relationships.
- Data Warehouse is a collection of rules for integral, objective and subjective strategic information which must be managed in all phases, starting from data acquisition process. The necessary characteristics that the data warehouse must own imply data modeling and usage of design techniques that support subjective orientation. It also provides data integration flexibility through additional data sources in all data live cycle.

The data warehouse is a historical store of strategic information with historical data relations with different entities. This property enables data modeling and easy usage of supporting design techniques for nonvolatile and time consistence. It is the data source from which the data is exported in the sector's Data marts. The sector's Data marts can be used for data exploration, data mining, and manager's reports or as a base for gaining multidimensional cubes for analytical data processing called OLAP cubes. This implied using a model of unbiased data which will be filtered and prepared to satisfy specific aims and demands of data marts. Some models for data aggregating support are being used and also aggregated tables are created because of the minimizing the response time and increasing the end users satisfaction.
Data warehouse is a source of stable and nonvolatile data, from the aspect of data processing and data changing. The model for this type of data must be different from the transactional data model [5], because when we load data in the warehouse, they become historical data and they can't be updated or deleted. They must be browsed or selected.

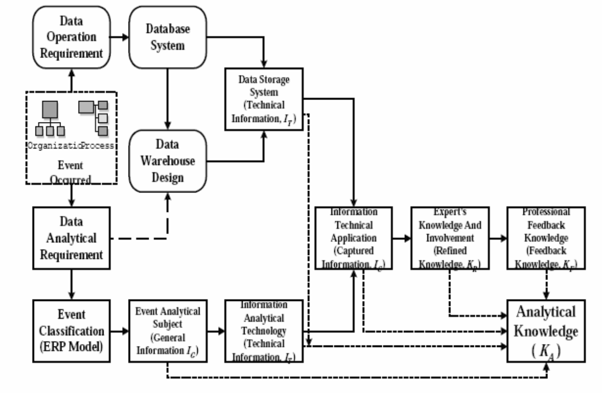

Data Operation Requirement
Database System
Data Storage System (Technical Information, I_T)
Organizatio Process
Event Occurred
Data Warehouse Design
Data Analytical Requirement
Information Technical Application (Captured Information, I_C)
Expert's Knowledge And Involvement (Refined Knowledge, K_R)
Professional Feedback Knowledge (Feedback Knowledge, K_F)
Event Classification (ERP Model)
Event Analytical Subject (General Information I_G)
Information Analytical Technology (Technical Information, I_T)
Analytical Knowledge (K_A)

Fig.2 Relationship of Data Warehouse with other company's IT systems [5]

3 Data warehousing and Data marts

The concept of data warehousing is a new one. For this reason, we need to standardize the definitions and the core issue, because in the literature we can meet different ways and definitions related with this issue. For this purpose, we usually use the company's data dictionary. Some issues are focused on data and other on people, software, tools and the business products. W.H.Inmon created the clear definition of the concept of data warehouse in the direction of measured attributes which clarify our understanding: "Data Warehouse is a collection of integrated, subject-oriented databases, designed to support decision support systems, which data is in volatile and relevant at the same time" (1992 Inmon).
Inmon's data warehouse definition make two implicit assignment:
(1) Data Warehouse is a physical place separated from the operating data
(2) Data Warehouse contains aggregated data and atomic analytical data for management, separated from transactional processing systems
Demands of separated data environment for data warehouse are an essential element of the concept. In many cases, the operating processing system is inconvenient for many business processes such as decision making and data analysis. This is obvious especially in the process of data visualization where data preparation time is required. In this case, data must be constant, integrated and time assigned.
First of all, we must explore some associated elements: How data will be stored in the company's data warehouse, how to make data marts and the data warehouse's metadata. The sources of data for the data warehouse - the transactional databases, are subject oriented, volatile and customer and product oriented, or focused on the operator' demands. Data transformation and integration in the data warehouse's counterpart or transformations of data events instances are not suitable for loading in the data warehouse. Transactional data has to be integrated with other unrelated transactional data, sets of internal or external company's data. There must be some procedures for data distillation and integration for the given flow of "loading" data in the organizational data warehouse. These procedures are created as scripts or triggers and run automatically in specified time, creating an updated data warehouse – refreshing data from data sources valuable for data warehouse.

Although the central aggregation concept, there is some kind of specific aggregated data which can be presented in same business organizational departments but is tangential to other departments. Alternative concept is adopting of scalable, less expensive version of data warehouse, called data mart. The data mart concept is sometimes called mini-mart and is posed to minimize the expenses and maximize the usefulness for organizational business departments. From the company's perspective, data mart can be an isolated island of information, accessible for top managers and the owner - organizational department which use this data mart. But, it isn't accessible for the other company staff. Because of these reasons, first it is convenient, to create data warehouse for the entire company with a complete view of the company and then organizational department's data marts. Another possibility is first to create the organizational department's data marts and then to integrate them, they create central integrated company data warehouse, depending of the manager's demands. This is more targeted and less expensive method of gaining DW from the existing data marts

The metadata is an important data warehouse characteristic; they are the information for data in the data warehouse. Each system must save the data attributes and the data transformation algorithm for the data warehouse, from where data is created, place where they are located, how are they transformed and how can be accessed. In this way, we obtain the possibility to crate a personal warehouse to meet specific customer needs. In fact, data warehouse is subjectively oriented, data integrated, time dependant and in volatile [1].

Creation of applicative data design of the data warehouse is always done with the defined variable names, the screen's variables and other issues are placed in the data dictionary of the data warehouse. All changes in the names and variables lead to inconsistency of applications with data dictionary and they must be automatically updated and changed in the organizational manual, but this is allowed until the moment when the data is loading in the data warehouse. In that moment, the discussion of which convention will be accepted and which names will be used, is finished. All the accepted conventions are saved in the metadata dictionary.

Another result of the data integration is retrieved of the common unit measures for all of the synonyms for the data elements in the data warehouse tables. Common unit measures integrate different measuring attributes and also create global acceptable units for all of the final data in data warehouse. For these reasons, there must be an organizational consensus for all of the data measures and all of the data units must be converted in these units when they are loading in the data warehouse. This agreement is available with creating a standard that will be accepted by the organizational staff and managers and will be used via all data warehouse's users. Sometimes the unit's unification must be obstruct and limited with a wide range of the company member's demands and demands for different data query with various unit measures. In this case, it is necessary to convert the data in a non-standardized unit measures and permitting the flexible staff's conversation setup. In the data visualization process, the retrieved of unified measures play an important role because of the fact that the unified measures of data give a valid visual data representation, and opposite, the visualization process is meaningless. Without a doubt, the data must be stored in data warehouses in an integral, global acceptable way defined with business rules. Figure 3 shows the organizational data flow for data warehouse.

Time period included in the data warehouses contains data from many years, opposite of the transactional information systems which store the data in short period of time, maximum - one year. Each primary key contains an explicit or implicit time elements (day, month, quartile, and year). In the presentation, time unit must be included with the primary key. When the transactional data is transformed, some implicit data element must be added and loaded at the end of the period and this time period must be exact and constant. Exception is made just when the data is incorrect or poorly transformed. Inhibit changing and updating data in data warehouse have sense because of its in volatile nature. These operations are allowed in the transactional databases, but in data warehouse, only the operations of loading and data browsing are supported.

Another difference between transactional databases and data warehouses is the normalized form and minimized redundancy of transactional data, opposite of demoralized form of the data warehouse. Data in the transactional databases are usually in the third normal form, with eliminated, derived data elements and aggregated data. Designers of Data warehouses don't care about redundancy because redundancy is recommended and it provides a rapid system response for reports, query and data visualization[1].

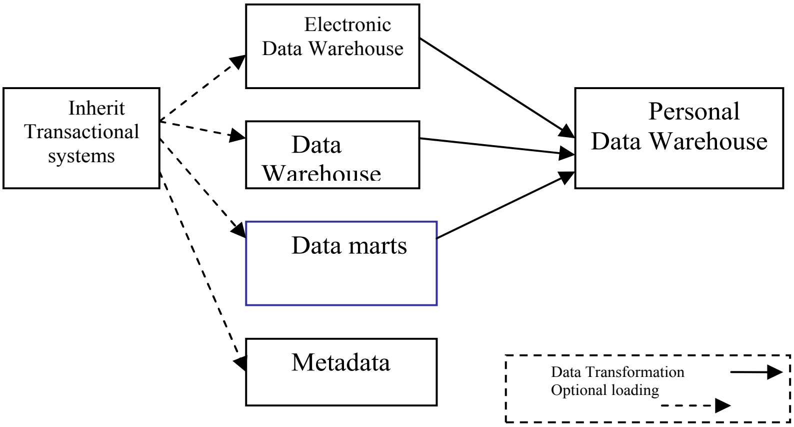


Fig. 3 Organizational data flow for Data Warehouse [1]

4 The Data Warehouse's Architecture

The data warehouse's architecture must support the organizational working data structure, communications and data processing. Figure 4 shows the relationships between data the warehouse's elements. The source level is presented by transactional database systems and external databases. The aim of the data warehouse is to release the data, locked in transactional information systems and to mix it with those from external databases. Additional aim of the data warehouse is to cause minimum influence to the system's operations to the data for analysis and processes monitoring. Web and internet technology allow easy and more economic access to data and incorporate the data in company data warehouse.

The metadata level contains data about data. When we insure universal access to data, some forms of directories with data and some data repository must be managed. Metadata includes a directory where data is stored with their names and mining, rules of summarization and data cleansing. In this directory some facts are provided, how data was prepared and how was stored in the data warehouse, from where data sources are taken and how they are saved.

The process management level is focused on planning tasks for creating and managing the data warehouse over data directories. For these reasons, this level leads us to create updates and managing procedures for data warehousing. Tasks as periodical downloading data from identified data sources, data aggregation and

[1] Inmon in 1992 show this and he was faced with the first impression was that there is minimal redundancy between transactional databases which are the sources of data and the data warehouse (around 1%). He enumerates couple of factors for this

downloading external data, as the metadata updates are allowed in this data warehouse architecture level.

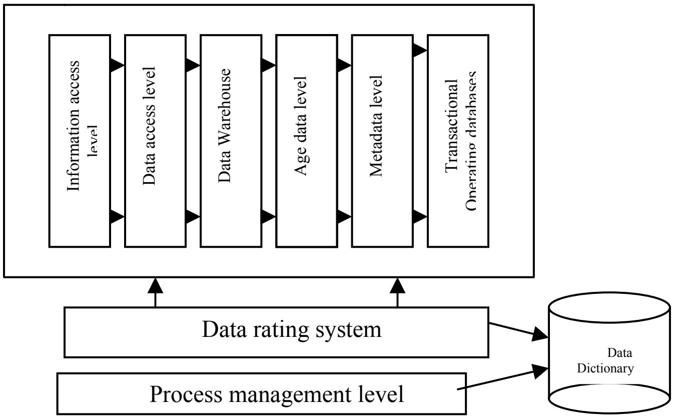


Fig. 4 Components of Data Warehouse Architecture [2]

Data access level transfers all of the information across the network. It is called middleware and includes network protocols and a routine of searching. Applicative sending messages can be used for some isolated applications. This level is also used to collect transactions or messages and deliver them to some locations at the same time. This level can be imagining as data warehouse message transport system.

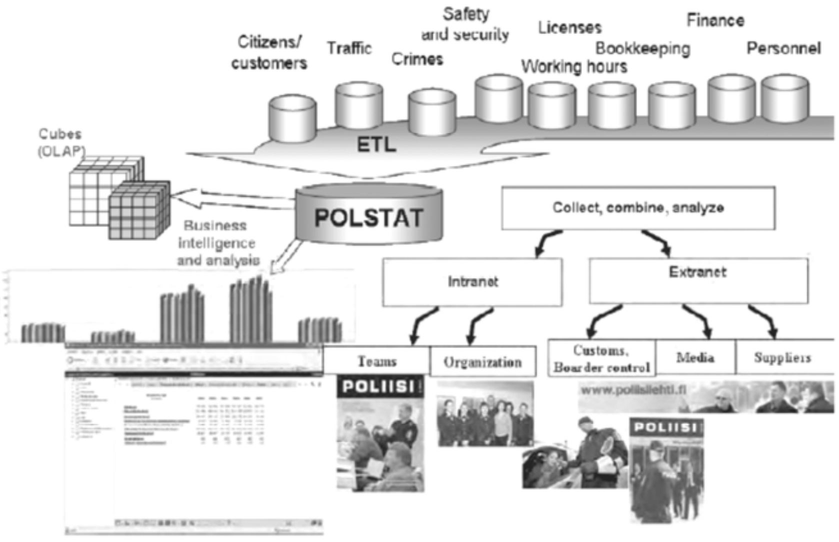


Fig.5 All DW levels must be adapted to business rules [5]

combining and data loading. Also, they include information about internal transactional company databases and external databases. The physical data level often includes complex programming. But, the data warehouse vendors reduce the process's complicity, making the tools easy to use. The demands of this level are analysis of data quality, filtering, model identifying and structuring data in the resent operational databases (Figure 5).
Another model to planning and development of the data warehouse architecture is a system that consists from six subsystems:
1. Operating data from transactional information systems which provide company's data for loading data in the data warehouse over application for extraction, data propagation, conditioning and data standardizing.
2. Data migration and collecting in data warehouse. In this case some form of synchronic and asynchrony conversion of data can be used[2]. Also, we can mention operations such as data refreshing, updating and propagation. In this phase, the processes of data restructuring as renormalization, adding new files, entities and keys, data combining and aggregating are also allow data to become information. There is data from internal and external sources, prediction data, assessments and simulations. The quality of this system is depends of the transport mechanism in the data warehouse such middleware.
3. Data warehouse designing and administration understands great knowledge for DW metadata, the relationships between them and their sources (internal or external). Also the designer must have information about the data relationships, their granularity, and period of updating, backup period. As the data warehouse becomes bigger, the more expensive it is to maintain it. Although the higher dimension of the data warehouse, the simplification of data is not recommended because of the essentiality of analytic data analysis and decision making. Today's data warehouse administration system includes usage of visual HCI interface which simplify data administration, data cleansing, data transferring and data loading (For example Oracle Data Warehouse builder).
4. The middleware is a system software for machine learning with purpose to collect and use knowledge of the abstraction level and it depends on the number and structure of the users, their connection via LAN or WAN in client-server environment[3].
5. Decision support and analysis applications demand creating database models, databases and application software which provide their connection and usage from the end users. The methodology includes different data access, mathematics and statistic models, visualization, heuristic models or using knowledge databases (or library) and case studies. Sometimes it is very hard to connect the solution complicity with the simplicity of the end-user solutions. For these reasons, such systems include data mining software, data exploration and sometimes data models in knowledge databases. Although all problems complicity, interfaces mustn't be overcrowded and with intensive color. The colors are welcomed for end users but sometimes, fear the user. This means that we can use reasonable screen visualization and understandable user interface.

[2] Detail in N.Balaban,J.Ristic, Sistemi za podrzavanje odlucianja, Saraevo, 1998
[3] This concept implied physical separation of applications and databases

6. Presentation interface is the most important system of data warehouse usage because it provides communication with users and courage or discourages them. Depending of its intention, the interfaces can be classified as:
a) Simple information interface in form of tables or visual presentation
b) Interactive systems with queries and drill-down possibility
c) Simulated system performing "what-if" analysis
d) Functionality systems that support corporative function providing company functions
e) Automated expert systems which help solving specific expert problems
Few of them may be combined in an unique interface. We can use interfaces such as: command prompt, menu-interface query language, graphics interface, groupware and multimedia and hypertext interface. The interface has to be easy for use and providing easy data access and data understanding for end-users.

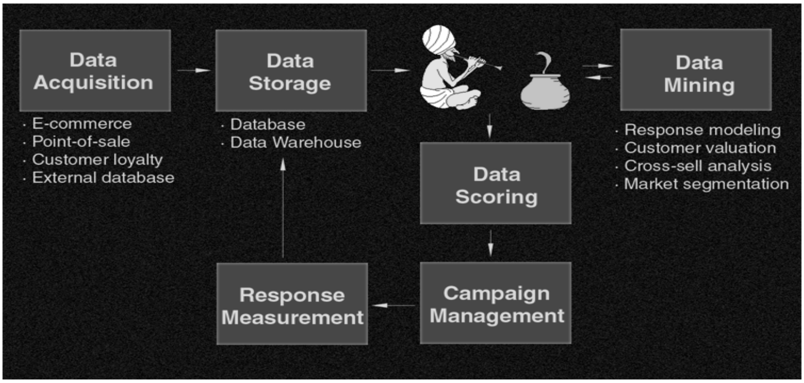


Fig. 6 Relationship between Data Warehouse and Data mining processes [4]

5 Conclusions

Data Warehousing is a preferable concept for all manager's and analytical demands because of the growing needs for data analysis and demanding information for decision support. For this reason, the data warehouse becomes focused on all organizational and informational needs. But, to build and maintain an organizational data warehouse it is not an easy process. There are some conditions to be satisfied, such as the company consensus for building warehouse, support from top managers and proper organizational behavior. Also, there must be support from experienced supporters for technical and software processes in the creating of the data warehouse. The most important thing is defining the business rules, granulation, metadata, data sources and model of data transformation. For these reasons, the data warehouse must be planned and created very carefully. We often think about data warehouse as a part of data mining processes in the company. The representation of the relationship of the Data Warehouse with data mining is shown on Figure 6.

References

1. Marakas D., “Modern Data Warehouse, Mining and Visualization”, Chapter 2 and 3, (2005),
2. Balaban N, Ristic J., “Sistemi za podrzavanje odlucivanja”, Saraevo, 1998
3. Inmon, W, H, “Building the Data Warehouse”, Third edition, WCP, 2007
4. Turban E., “Decision Support Systems and Intelligent Systems”, 2008
5. Wang J., Data Warehousing and mining, Second edition, USA, 2008

Multi-Source Customer Identification

Ina Naydenova, Kalinka Kaloyanova, Stoyan Ivanov
"St. Kliment Ohridski" University of Sofia, Faculty of Mathematics and Informatics
Sofia 1164, Bulgaria
ina@fmi.uni-sofia.bg, kkaloyanova@fmi.uni-sofia.bg

TechnoLogica Ltd.,3 Sofiisko pole Str.,1756 Sofia, Bulgaria
sivanov@technologica.com

Abstract. Data warehouse systems integrate divergent information from various systems which enable users to quickly produce powerful ad-hoc queries and perform complex analysis. In this paper we present a heuristic method for customer's identification. The method merges customers' information and eliminates duplicated objects across several operative systems. It has been successfully used in a real-life system from more than three years. We also discuss some common problems that we come across during the system life cycle.

Keywords: customers, identity, heuristic, merge, duplicates.

1 Introduction

A data warehouse is a subject-oriented, integrated, time-variant, nonvolatile collection of data [1]. Information integration is one of the most important and problematic aspects of a Data Warehouse [2]. When data passes from the sources of the application-oriented operational environment to the Data Warehouse, possible inconsistencies and redundancies should be resolved, so that the warehouse is able to provide an integrated and reconciled view of data of the organization [2]. Anomalies and impurities in data cause irritations and avert its effective utilization, disabling high performance processing and confirmation of the results and conclusions gained by data interpretation and analysis [3]. As a result, business intelligence systems experience low confidence and acceptance by users and consumers of downstream reports. Additionally, in many cases data warehouse projects have failed [4].

During the development of customer information systems for the needs of a group of Bulgarian companies we have to solve the object identity [5] and the merge/purge problem [6]. For this purpose, we use a heuristic method, conformed to specific anomalies and impurities of the source system data[4]. The method defines numerical distance estimation between similar objects and use a neighborhood based approach to identified clusters of identical customers. In the present paper we will describe the

[4] This paper is supported by the Sofia University SRF under Contract N 7/2009.

applied approach and method stages: data cleaning and preprocessing, identical candidate's collection, tuple estimation. The short discussion of common problems that we come across during the system life cycle is also presented.

2 Data Cleaning and Entity Matching Problem

Data cleaning deals with detecting and removing errors and inconsistencies from data in order to improve the quality of data. When multiple data sources need to be integrated, e.g., in data warehouses, federated database systems or global web-based information systems, the need for data cleaning increases significantly. In order to provide access to accurate and consistent data, consolidation of different data representations and elimination of duplicate information become necessary.

Data warehouses require and provide extensive support for data cleaning. They load and continuously refresh huge amounts of data from a variety of sources so the probability that some of the sources contain "dirty data" is high. Furthermore, data warehouses are used for decision making, so that the correctness of their data is vital to avoid wrong conclusions. For instance, duplicated or missing information will produce incorrect or misleading statistics ("garbage in, garbage out"). Due to the wide range of possible data inconsistencies and the sheer data volume, data cleaning is considered to be one of the biggest problems in data warehousing. Classification of the major data quality problems to be solved by data cleaning and data transformation could be seen in fig 1 [7]. The schema level problems are related with schema design, schema translation and integration, while instance-level problems refer to errors and inconsistencies in the actual data contents which are not visible at the schema level.

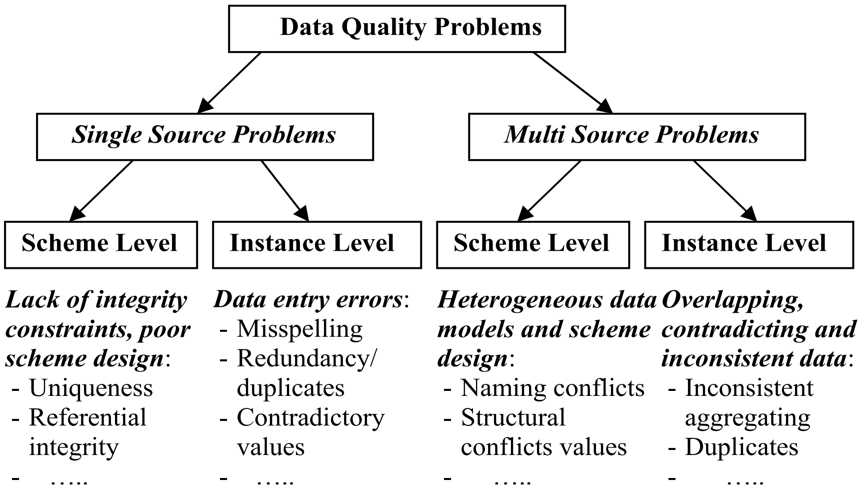


Fig. 6. Classification of data quality problems in data sources

A main problem for cleaning data from multiple sources is to identify overlapping data, in particular matching records referring to the same real-world entity (e.g., customer). Thus duplicate information should be purged out and complementing information should be consolidated and merged in order to achieve a consistent view of real world entities.

Customer (source 1)

CID	Name	Phone	Sex
708	Dimitrina Hristoforova	02/956115	0
102	Sirakov Maxim	062/34580	1

Client (source 2)

Cno	First Name	Last name	Gender	Mail Address	Phone
25	MAXIM	SIRACOV	M	5000 Veliko Tarnovo	062-34580
708	Dimitra	Hristova	F	Veliko Tarnovo	062-75129

Cusromers (integrated target with cleaned data)

No	FName	Lname	Gender	Post. code	City	Phone	Cno	CID
1	Maxim	Siracov	M	5000	Veliko Tarnovo	062/34580	25	102
2	Dimitrina	Hristoforova	F			02/956115		708
3	Dimitra	Hristova	F	5000	Veliko Tarnovo	062/75129	708	

Fig. 7. Examples of multi-source problems at scheme and instance level

The two sources in the example of Fig. 2 are both in relational format but exhibit scheme and data conflicts. At the instance level, we note that there are different gender representations (“0”/”1” vs. “F”/”M”) and presumably a duplicate record (Maxim Siracov). The latter observation also reveals that while Cid/Cno are both source-specific identifiers, their contents are not comparable between the sources; different numbers (102/25) may refer to the same person while different persons can have the same number (708); the third table shows a possible solution. Solving these problems requires both scheme integration and data cleaning; Note that the scheme conflicts should be resolved first to allow data cleaning, in particular detection of duplicates based on a uniform representation of names and phones, and matching of the Gender/Sex values.

The various approaches to entity matching proposed in the literature can be classified into two broad categories: rule-based and learning-based. In most of the rule-based approaches, the entity matching rules involve a comparison between an overall similarity measure for two records and a threshold value. The overall similarity measure is usually computed as a weighted sum of similarity degrees between common attributes of the two records. The weight of each common attribute needs to be specified by domain experts based on its relative importance in determining whether two records match or not. In rule-based approaches, domain experts are required to directly provide decision rules for matching semantically corresponding records. In learning-based approaches, domain experts are required to provide sample matching (and non-matching) records, based on which classification techniques are used to learn the entity matching rules [8].

According to this classification our method is related to rule-based approaches. The entity matching rules involve a comparison between important attributes for two entities; if the similarity is above the specific threshold, the two records are considered matching. An attribute preprocessing is applied to derive compatible common attributes by converting the formats and cleaning anomalies of originally incompatible attributes. But, on the other hand, our attribute estimation and threshold value are initially accommodated to the sample (matching and non-matching) records extract.

2.1. Customer information system

Our customer system integrates data from several companies with different type of business. The system collects data for heterogeneous data sources through flat files transfers with common structure. In such way the problems with schemes integration and different attributes representations are resolved during a source systems data extraction. The data for companies' customers is loaded every day in the system. The object identification and entities matching process is critical for the system working. All analyses made by the system end users are related with the customer profile – customer service consumption, different types of financial metrics, key performance indicators as average revenue per customer, new customers acquired, customer segmentation and etc. So without a duplicate elimination the results will be very wrong and unreliable.

In table 1 you can see the percentage of the single-source duplicates elimination in the system. As could be expected the values are small, even negligible (every source systems tried to support a customer register without object duplicates). But the percentage is much bigger when we consider the identification between sources. The multi-source duplicates elimination percentage is 55.82 %.

Table 2. The percent of the single-source duplicates elimination

Data Source	Percent
Bank Company Source 1	12.40%
Bank Company Source 2	0.61%
Insurance Company Source 1	13.48%
Insurance Company Source 2	0.42%
Insurance Company Source 3	18.24%
Insurance Company Source 4	10.73%
Insurance Company Source 5	13.25%
Life Insurance Company Source 1	1.07%
Life Insurance Company Source 2	1.85%
Pension Company Source 1	0.21%
Pension Company Source 2	0.05%
Pension Company Source 3	8.45%

3 Customer's identification framework

The customer identification process that we use has three main phases: data preparation and preprocessing (CIA_PREPARE), identical candidates pairs collection (CIA_SEEK) and candidates evaluation (CIA_EVALUATION). Let's see in a little more details what we do during every phase.

3.1 CIA_PREPARE

This stage envelops activities such as splitting of unformatted data and fields preprocessing (transformations and validations procedures). The goal of this phase is to derive new attributes values that will be used in the next two phases.

In our system we distinguish two types of customers – private persons and companies (pp and co.). The relevant attributes used in identification algorithm for private persons are person name, contact mail address, telephone number, identification card data, personal identification number (personal identifier), birth date and gender. For organizations we are interesting in their name, contact mail address, contact telephone number, identification code (Bulstat code), legal registration and tax reference.

As indicated in Table 2, firstly we do some transformations: eliminate common and therefore less meaningful words that could bring an unnecessary noise in data (for example Limited, Plc, Corporation, systems, Inc., Miss, Mrs, etc.) from the name attribute, and disjoin the separate components in the mail address, identification card and legal registration;

Table 3. Attributes transformation rules

Attribute	Transformation
Name (for pp and co.)	Common words, redundant spaces, dashes, points, slashes and special symbols (&, and) are eliminated. All letters are converted in uppercase. Latin symbols with the same visual presentation as Cyrillic ones are replaced (for example A, E, T, M etc.). First, surname and family name components are derived.
Address (for pp and co.)	The postal code and settlement components are derived (where it is possible).
Identification card (for pp only)	The card number and date of issue are derived (where it is possible).
Legal registration (for co. only)	The number of the registration case and year are derived (where it is possible).

Then we prepare an extract of the customer name. The goal of the extract is to eliminate those parts of the name, which are target of common sound proximity mistakes (see Table 3). For example the extract on the name “Ina Asenova Naydenova” produces three components “In”, “Asnv” and “Ndnv”.

Because the birth date and the gender of the private persons are coded in their personal identification number, we also extract these characteristics in separate attributes. This rule is not true for the foreign citizens in Bulgaria. Before the extraction we check the validity of personal identifier. We also recognize it as an identifier of Bulgarian or foreign resident.

Table 4. Extracts transformation rules

Extract	Transformation
Single name extract procedure	1. The first letter is preserved; 2. The contiguous equal letters are replaced with only one copy; 3. All vowels, separators and special symbols are removed.
Compound name extract procedure	The single name extract procedure is applied to every component of the composite name (components are words separated by blank). The results are stored as an unordered set.
Birth date extract	Extract the first 6 digits from the personal identifier
Gender extract	If the 7th digit of personal identifier is odd then the person is a female.
Phone number	Only the digits from the attribute are extracted

The third step of the CIA_PREPARE consists of some validations. These validations are not very strong. For example, there is a formal procedure for Bulstat code validation. We use it to issue an alert for invalid Bulstat code detection when data are loaded in the system, but for the purpose of customer’s identification we use a quite simple validation – if the number has not more that 6 repeated digits it is good for identification. Many of our validations rules are based on the statistical information derived from the source data. The goal of this step is to eliminate artificial and non-identifying attribute values, but to preserve object specific values even if they are not completely correct (table 4).We use an attribute in the next phases only if it has passed the validation procedure.

Table 5. Attributes validation rules

Attribute	Validation
Personal identifier	Is valid if it is a valid foreigner number according to the official validation criteria or it has no 6 repeating digits, contains no 4 repeating zeros (0000) and a birth date component derived from the first 6 personal identifier symbols is a valid date.
Bulstat code	Is valid if it has no 6 repeating digits
Tax reference	Is valid if it not have 6 repeating digits and it has exactly 10 digits

Identification card number	Is valid if the size of the card number component is more than 8 symbols, it does not have 6 repeating digits and it does not contain the following digits sequence:123456
Birth date (according to input data)	Is valid if it is a valid date
Birth date (derived by the personal identifier)	Is valid if it is a valid date and the personal identifier passes the official validation criteria
Gender (according to input data)	Is valid if it is one of the codes F (female) or M (male)
Gender (derived by the personal identifier)	Is valid if the personal identification number passes the official validation criteria
Customer type	Is valid if it is one of the codes P (person) or C (company)

3.2 CIA_SEEK

During the second phase the available data is ransacked for very likely identical objects. Because the number of possible pairs when we deal with 5 million customers is $25x10^{12}$, if we estimate every pair this will be very slow process. The goal is to form a reduced list of pairs which will be precisely estimated in the next phase. The criterion that we use to make up our candidates list is the following: If the customers A and B have overlapping values according to the personal identification number, Bulstat code, identification card number, tax reference or name, then this pair (A,B) is included in the list. The symmetrical pair (B, A) is not included.

3.3 CIA_EVALUATION

The goal of the last phase is to estimate the similarity measure of every pair in our candidates lists, computed as a weighted sum of similarity degrees between attributes. The similarity measure is an integer number. We give specific scores when the values of a particular attribute are equals for a candidate pair. The scores magnitude is defined on the base of heuristic grade of the attribute importance related to the objects identity. We also give a penalty (negative scores) when some key attributes are different. Table 5 indicates the number of scores that we give for equal and non-equal attributes. Two customers are considered to be identical when the sum of their scores is more than 250. The system also allows customers to be explicitly declared as identical or not identical. For such pairs we give 999, respectively - 999.

Table 6. Similarity degrees between attributes

	Attribute	Similarity score
Equal	Personal identifier	+200
	Bulstat code	+300
	Tax reference	+300
	Identification card number	+300
	Birth date (according to input data or derived from the personal identifier)	+50
	Name	+90
	Phone number	+30
	The settlement (from the mail address)	+20
	The postal code (from the mail address)	+20
Non-equal	Gender (according to input data or derived from the personal identifier)	-150
	Customer type	-200
	Name extracts (if there are less than two matching extracts and each of the compared customers has at least two extracts)	-50
	The settlement (from the mail address)	-50
	The postal code (from the mail address)	-20

Since it is possible for a customer A to be identical with B, and B to be identical with C, after the pair evaluation we apply an algorithm to discover the transitive closure of the identity relation.

We preserve not only the result of customers' identification, but also the entities that form a unified customer profile. When a new portion of customer data is loaded or updated in the system, the candidate pair list is formed with comparison of newly loaded or updated customers, versus already loaded original source (non-identified) customers. The unified customer profile contains the most actual attribute values. It is possible for the value of one attribute to be received from one source and the value of other attribute to be taken from a different source.

3.4 Common Problems

We will discuss some problematic cases that we came across during the testing and the life cycle of the systems. Most of them impose a few corrections so the described rules to be accommodated to the real life data:

Attributes preprocessing

We notice that the Bulgarian letter "З" (Latin Z) sometimes is replaced with the digit three in the customer's names. In the same way the letter O is replaced by the digit zero. Also, we found cases when private person names are composed by two different names and the description of their relation. For example: "X and Y - mother and son". So we had to enlarge the list of symbol replacement. The compound name extract procedure derived components for the 2 person names. We do not know who of these persons the right one is, but since the equal name is not enough for party identification, we still give scores if one of the names is equal to the name of identical candidate.

One of the more serious problems is the case when the operator, who enters the customer data in the source system (for example the insurance agent), does not know some customer characteristics (for example the identification card number, personal identifier etc.) and he/she enters his own data. In such case the identification algorithm brings together all the customers that are served by the system operator. Later he/she corrects the data with the right ones. But for our system this new data are treated as changes in attribute values to the same source customer and the wrong identifications that are already done are not erased.

Candidate pair lists

We notice that a large number of pairs are included in the list because of the popularity of the some names (Maria Dimitrova, Georgi Ivanov etc.), but these people are completely different and they do not have a chance to pass the identification threshold. So we exclude them from the candidate list.

4 Conclusions and future directions

In the present paper we describe a framework and rules for customer identification that we have implemented. It is successfully used in a real-life customer information system in the recent years. But our method has limitations that need to be addressed in future. The customer identification is the one of the process direction. The other side of the process is the ability for automated splitting of identified customers when the wrong input data is detected. The transformations could be integrated with the statistical and learning methods in order to analyze the dirty data and help the user to find data problems that need to be cleaned. In the current framework we use several types of reports that show the suspicious identifications – for example a bundle of more than 6 equal customers, identified customers with big variety in their attributes, big variety in service consumption etc. We also report warnings during the data loading, but they are preliminary implemented and are not capable to mutate dynamically. We look forward for further enhancement and development of our customer identification framework.

References

1. Inmon, W. H.: What is a Data Warehouse. PRISM Newsletter, Center for the Application of Information Technology, Washington University in St. Louis, vol. 1, no. 1 (1993).
2. Calvanese, D., De Giacomo, G., Lenzerini M., Nardi, D.,Rosati, R.: Data integration in data warehousing, International Journal of Cooperative Information Systems, 10(3), 237--271 (2001).
3. Muller, H., Freytag, J., Problems, Methods, and Challenges in Comprehensive Data Cleansing, Technical Report HUB-IB-164, Humboldt University Berlin, Germany, (2003).

4. Linsley, S., and Dutta, A., The Next Frontier for Data Warehouse Managers, DM Review magazine, February, 2008.
5. Galhardas, H., Florescu, D., Shasha, D., Simon, E.: Declaratively cleaning your data using AJAX. In Journees Bases de Donnees, (2000).
6. Hernandez, M. A. and Stolfo, J. S., Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem, Journal of Data Mining and Knowledge Discovery, vol. 2, pp. 9-3, (1998).
7. Rahm, E.,Do, H.H.,Data Cleaning: Problems and Current Approaches, IEEE Techn. Bulletin on Data Engineering, Dec. 2000, p.11, (2000)
8. Zhao,H.,Ram,S.,Entity matching across heterogeneous data sources: An approach based on constrained cascade generalization, Data & Knowledge Engineering, v.66 n.3, p.368-381, (2008)

Modeling of Distributed Transactions with Priorities

Svetlana Vasileva

Shumen University “Bishop Konstantin Preslavski”, College – Dobrich. Dobrotica 12, Dobrich, 9302, Bulgaria
svetlanaeli@abv.bg

Abstract. One of the major problems in Distributed database management systems is concurrency control. The paper considers an approach for modeling management of the transactions parallelism with the priority to distributed database management systems (DDBMS). As deadlocks originate in concurrency control algorithms, based on two-phase locking (2PL) protocols, in order to solve this problem, we offer priority service of the transactions. However, the use of priorities makes the algorithms for transaction management more complex due to the removing of the transactions from service and the additional waiting for processing. This is the reason we consider a modeling algorithm of centralized 2PL in greater details.

Keywords: distributed transactions, distributed databases, concurrency control, simulation models, priorities, GPSS blocks.

1 Introduction

In Distributed Database Management Systems (DDBMS) as a result of fragmentation and data replication, the reliability and throughput of the systems is increasing. One of the major problems in this kind of systems is the concurrency control to the shared data of the executed transactions. The distributed transaction accomplishes the access to the data, which are stored in different places of the distributed database system.

Commonly, each of the transactions is divided into several subtransactions, one per each node to the data to which an access is realized. DDBMS with data replication should supply sequence of the many copies stored in various nodes of the system. This work considers the synchronic update of the replicated data elements according to the two-phase locking in DDBMS methods. These protocols give us better results than the optimistic methods in systems full with conflict events. This is the reason we choose to view an approach for developing modeling algorithms for management of the transaction parallelism with priority, which are based on 2PL protocols in distributed databases (DDB). As a basis for this development, we use the presented and considered in [4], [5], [6] and [7] simulation algorithms of centralized 2PL, 2PL with primary copies, distributed 2PL and voting 2PL. The results of the conducted simulations are shown in these works. Data in the modeling algorithms are exposed to incomplete replication (all data elements have one and the same number of replicas, the number of copies of an data element is smaller than the number of the sites).

The simulation models are developed with the means of a “classical” system for modeling – GPSS World.

As in the pessimistic protocols, it is possible deadlocks of transactions to appear there, we have come across some problems concerning the detecting and solving the deadlocks. One of the ways to avoid them is to nominate priorities of the transactions. In [1], [2] and [3] are suggested algorithms for 2PL in Real-Time Database Management Systems (DBMS): 2PL-HP (High Priority) and 2PL-WP (Wait Promote). For 2PL-HP it is claimed that it guarantees the absence of deadlocks.

2 Two-Phase Locking Algorithms with Priorities

Two protocols for two-phase locking are known [1], [2] and [3] and they solve conflicts for a locking in favor of the transactions with higher priority.

2.1 2PL-HP (High Priority)

The main idea of the protocol is all conflicts to be solved in favor of the transactions with higher priority. When a transaction *T* requests the locking of data element, then it follows that:

- If the element is free, *T* can put the locking of the element;
- If the element is locked by another transaction, then it follows that:
  - if *T* wants to receive the lock and it is not in conflict with locks of the element of other transactions which are already put, *T* will get the locking only if its priority is higher than that of the other transactions which are waiting the locking of the element;
  - if *T* has a higher priority than the transactions which hold the element locking, these transactions have to be interrupted and *T* has to get the locking;
  - otherwise *T* queues in this locking.

2.2 2PL-WP (Wait Promote)

When *T* transaction with higher priority requests the data element locking, which in this particular moment is locked by a transaction with lower priority, *T* queues for the locking, but the priority of the transaction that holds the locking increases to the level of the *T* priority. As a result this transaction may be executed faster (because it will wait less for other data elements), hence the waiting of *T* for the locking will be shortened.

Both of the protocols described here are used in Real-Time DBMS. The analysis of the advantages and disadvantages we made says that we can use some positions of the two protocols, and mostly those of 2PL-HP for elaboration of the management of the parallelism of the distributed transactions in DDBMS.

3 Basic Elements of the GPSS Simulation Models

The presented simulation model uses generated streams of transactions which imitate global transactions in DDB systems. They are all in parallel streams and their intensity λ is given in tr per sec (number of transactions per second). The modelling of the distributed transactions in DDBMS with GPSS transactions and the resolutions for 2PL modeling in DDBMS with data replication are described in detail for each of the four protocols in [4], [5], [6] and [7].

The structural scheme of a modelling algorithm for distributed transactions management with a priority is shown in fig. 1. This work suggests the modelling of distributed transactions with priorities and the concurrency control in GPSS World environment according to an already mentioned scheme.

3.1 Parameters of the GPSS transactions

P1 – Number of transaction. The value is a sum of System Numeric Attribute MP2 (The subtraction between the relative model time and the content of the second parameter of GPSS transaction) and the number of the site;

P2 – Number of the site, where the transaction is generated. The value is a number from 1 to <number of stream transactions>;

El1 – Number of the first element, which the generated transaction will read or write. The value is a random number and is uniformly distributed in the interval [1, NumEl];

El2 – Number of the second element, which will be processed by the generated transaction;

Bl1 – Type of the requested lock for the first element, which will be processed by the generated transaction;

Bl2 – Type of the requested lock for the accessed second element;

P5 – Value 0, if the transaction is in 1st phase – occupation of the locks and value 1, if the transaction finishes its work and has to release the locks. In the future value 2 will be able to be appropriated to a transaction which will have to be restarted;

P6 and P7 – In them there are correspondingly recorded the number of the site, where it is the nearest copy of the data element and the number of the site, where it is the second replica of the first data element, processed by transaction. Correspondingly in parameters P8 and P9 we have the nearest copy of the second data element and the number of the recorded site, where it is the second replica of the second data element. The transfer through the network to the central lock manager LM_0 and to the sites-executors, where are the data managers DM is simulated with retention.

3.2 Basic operations

The requests for locking of the data elements l_i wait for the release of the lock manager (LM) in the queue before it QLM_0. They queue firstly according to a priority and secondly according to the order of getting there, thirdly according to the number of the site – generator.

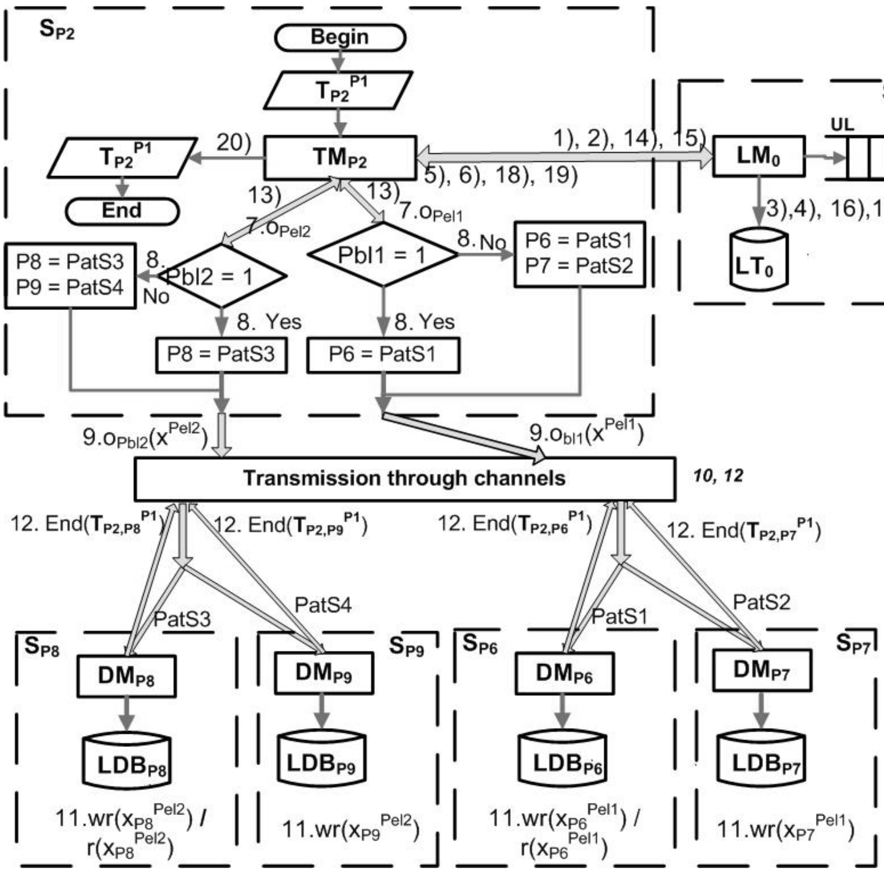


Fig. 8. Scheme of Centralized 2PL of transactions with priorities. Significations: 1) and 2) – $l_{Pbl1}(x_{Pr}^{Pel1})$ and $l_{Pbl2}(x_{Pr}^{Pel2})$; 3) and 4) – Rec<$l_{Pbl1}(x^{Pel1})$> and Rec<$l_{Pbl2}(x^{Pel2})$>; 5) and 6) – granted_$l_{Pbl1}(x_{Pr}^{Pel1})$ and granted_$l_{Pbl2}(x_{Pr}^{Pel2})$; 8) Check and preparation for (non)split of the corresponding subtransaction; 13) End(T_{P2}^{Pel1}) and End(T_{P2}^{Pel2}); 14) and 15) - $ul_{Pbl1}(x_{Pr}^{Pel1})$ and $ul_{Pbl2}(x_{Pr}^{Pel2})$; 16) and 17) - Rec<$ul_{Pbl1}(x_{P1}^{Pel1})$> and Rec<$ul_{Pbl2}(x_{P1}^{Pel2})$>; 18) and 19) – confirm $ul_{Pbl1}(x_{P1}^{Pel1})$ and $ul_{Pbl2}(x_{P1}^{Pel2})$; 20) Done(T_{P2}^{P1}).

With the suggested algorithm our aim is to avoid the appearance of deadlocks. In a distributed environment their resolution is a complex and labour-consuming process. As a result of the forced waiting of the transactions with lower priority, it is assumed that the average time of response of the system will increase. But this increase will be smaller than that if the transactions do not have a priority and if there is a forced waiting until the discovery of the deadlocks.

The operations concerning the service of distributed transactions in the model are:

- After the coming of the global transaction, it splits into subtransactions corresponding to the elements which the transaction will process - operations 1) and 2) on fig.1;

- If the transaction processing El1 receives the locking of the element, it puts the corresponding record in Lock Table LT_0 (the matrix LTA) – operation 3) on fig.1; Analogically – for the subtransaction for El2 – operation 4) on fig.1;
- If the locking of El1 is granted – operation 5) on fig. 1, the subtransaction should execute the corresponding operation read/write, defined by the parameter Pbl1. If Pbl1=1 (read operation), the subtransaction for El1 splits only once and then continues its way towards the data manager DM_{P6}, if Pbl1<>1, subtransaction for El1 splits twice and the two transactions continue their way towards the corresponding data managers DM_{P6} and DM_{P7} – operations 5), 7) 8), 9) and 10) on fig.1. Analogically, we have the operations for the subtransactions for El2 – operations 6), 7), 8), 9) and 10) on fig. 1. Operations, which are executed in parallel in the model have one and the same number in the scheme of fig.1;
- In the executive sites, where the corresponding DM are there are fulfilled the operations read/write data element. This is modeled through detention. In the model the segments with PatS1 and PatS3 labels fulfil operations read/write over the first copy of El1 and El2 correspondingly and the segments with labels PatS2 and PatS4 do only operations write on the second replica of El1 and El2– operations 11) on fig.1;
- After finishing the work of the subtransactions in site-executors, they are forwarded on the net to the central lock manager LM_0 (operations 12 on fig.1), in order to release locks of El1 (operation 14) and El2 (operation 15 on fig.1). Before releasing the locks of El1 and El2, subtransactions, which read/write, they assemble (operations 13 on fig.1);
- The release of the locks of El1 and El2 is simulated through nulling the corresponding lines in LTA matrix (operations 14 and 15 on fig.1);
- If there are waiting subtransactions for the locking on El1 and El2 transactions, they are removed from the corresponding user list through GPSS block ULINK and they put the corresponding records in LTA matrix – operations 16 and 17 on fig.1;
- After releasing the locks on El1 and El2 the subtransactions come back to the generator site, which is a confirmation of releasing the locks of El1 and El2 (operations 18 and 19 on fig. 1);
- After arriving at the generator site the subtransactions assemble and leave the system – operation 20 on fig.1.

3.3 Process of Transactions with Priorities

For the purpose of the modeling algorithm for management of the distributed transactions with priorities in 2PL protocols GPSS World environment it is suggested that:

- In the function description section a new function *Prio* to be added in which is described the probability with which the generated GPSS transaction will have a particular priority. The model stipulates four levels of priority. For the purposes of specific DBMS it could be given more or less number of priority levels.

- In the generating of the transaction streams in the *GENERATE* blocks in field *E* is given the name of the function which gives the priority of the GPSS transaction – *FN$Prio*.
- In positioning of the devices for transaction service (mainly in modeling the work of the lock managers) instead of the couple of blocks *SEIZE – RELEASE* it is used the couple *PREEMPT - RETURN*. In the model the block *PREEMPT* does not work in a regime for GPSS transaction elimination. Moreover, it waits that the GPSS transaction which has taken the device finishes the service.
- From the queues for waiting before the devices for transaction services, we choose the GPSS transaction with the highest priority and if they are several in number we choose the earliest one that has come in the queue.
- As in general, the first and the second elements are not stored in Local Database (LDB) on one site, two new segments PatS3 and PatS4 are added and they are used for defining the way of the splitted GPSS transactions accessed to the replicas of second data element.
- We provide user lists UL, where transactions wait for the release of the data elements.
- When a (sub)transaction is in a second phase (i.e a read/write operation was done, and it should release the locking of the element, it releases the first subtransaction which is waiting in the user list. If it is a subtransaction processing second data element El2 (i.e the released transaction is coming to its end) of the subtransaction, it is given a higher priority.

4 Simulation

Below we give some fragments of a program GPSS World code, which are specific for the distributed transactions modeling and their service with priorities. We view a model of Centralized 2PL as an easier one for the realization of the mentioned four algorithms in [4], [5], [6] and [7].

Example of a Computer Program from Vasileva S. (2008) GPSS World model of Centralized 2PL with priorities

```
**Program Centr2PL60PR.GPSS
LockTip FUNCTION RN3,D3   ;probability of coming up
.40,1/.70,2/1.0,3    ;of the particular type of locking
DistrS1 FUNCTION V$SiteRepl,D6 ;Replication of the data
1,2/2,6/3,1/4,5/5,3/6,4     ;First node for the element
… ;Replication of the data -Second node for the element
…
Prio FUNCTION RN5,D3

.30,1/.60,2/.80,3/1.,4
; Description of the used matrices
; Description of the storages
; Description of the tables for statistic
*** Segment for transaction generation
```

```
        GENERATE 60,FN$XPDIS,,,FN$Prio  ;generation of
Potok1  ASSIGN 2,1        ; GPSS transactions with the
        ASSIGN 1,(MP2+FN$NomSait)    ;given incoming
        TRANSFER  ,BEGI          ;interval (60 ms)
…
        GENERATE 60,FN$XPDIS,,,FN$Prio
Potok6  ASSIGN 2,6
        ASSIGN 1,(MP2+FN$NomSait)
BEGI ASSIGN El1,V$ElemN      ;Number of the Element1
     ASSIGN Bl1,FN$LockTip  ;Lock type of the Element1
     ASSIGN 5,0       ;Flag for fixing the transaction
…;Number of the Element 2. Check if the numbers of
Element 1 and Element 2 overlap. If they do, E2 gets a
new random number.
     ASSIGN Bl2,FN$LockTip  ;Lock type of the Element2
…
… ;counting of comited transactions
; Processing in transaction coordinator. If the T would
; be split into subtransactions to execute the read or
; write operations, check up and defining the number of
; the site of the first copy of El1 and defining the
; number of the site of the second copy of Element1)
; Parameters P8 and P9 get values
Pered  ADVANCE MX$RAZST(P2,1),MX$RAZDEV(P2,1) ;Transfer
;in the net to the LM0. waiting in front of LM0
LockR    PREEMPT LM0,PR     ;processing in the LM0
       RETURN LM0
       TEST E P5,0,Fiksira
       TEST E CH*El1,0,Chakane
Zaema1  TEST E MX$LTA(P$EL1,1),0,ProvR  ;"Is Element1
… ;free", if "Yes" then the Element1 from LT is taken
       TRANSFER ,Cepi
ProvR1   TEST E V$RAZRBL1,1,Chakane1  ; If the element
; is taken by a reading Tr and the transaction wants to
; read it, it will receive the locking for reading in
        TRANSFER ,Cepi1      ; The reading transaction
 is heading to the data manager
Chakane1 LINK P$El1,FIFO      ; Waiting for the
;  transaction "in a queue" before an element
Cepi1   SPLIT 1,PatS1,REPLI   ; the first subT
        SPLIT 1,PatS2,REPLI      ; and the second
;   subtransaction is heading to the corresponding DM
Cepi01  TRANSFER ,Obd1
Fiksira1 …      ; the first subtransaction releases El1
*****Release the lock of the 2nd element
Fiksira2 TEST E P$Bl2,1,RelLock2    ; If the
; subtransaction has only read El2
…
```

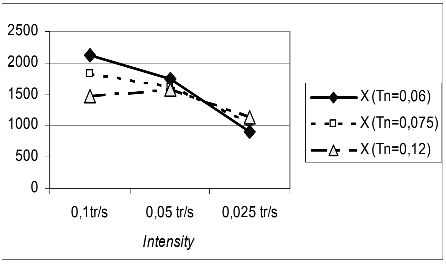
```
PrvChet2 TEST NE MX$LTA(P$El2,4),P1,RelLock2
         TRANSFER ,Krai
RelLock2 … ; Nulling the lines for El2 in LT
** if the queue before El2 is not empty
        TEST G CH*El2,0,Fik2
        UNLINK P$El2,Provch02,1 ; the first one which
; is waiting is released
Fik2     TRANSFER  ,Krai  ; the subT which has
; finished its work heads towards the site generator
Provch02  PRIORITY 6  ; acquisition of a higher
; priority of the released subtransaction
        TRANSFER ,Zaema2
…
Krai ASSEMBLE 3     ; assemble the subtransactions
    ADVANCE MX$RAZST(P2,1) ; to the site-generator
…
```

5 Simulation Results

Our research has been made for 2 replicas of the element, number of the incoming streams - 6 and number of the data elements in the global database - 50. Several parameters have been changed in order to obtain the whole picture of the states in which the chosen model went through.

The parameters and indexes of the simulations of the considered model are as follows: NumTr – general number of the generated transactions for the time of incoming modeling; FixTr – general number of the completed (committed) transactions for the same period; X=FixTr/Tn – throughput of the queuing system; Tn – time interval in which the system is being watched; Ps=FixTr/NumTr – probabilities for transaction service. The results are received in 6 streams of concurrent transactions with different intensity. The copies of the data elements are distributed evenly and random by 6 sites in the system. Service and rejection probabilities are calculated according to closely associated formulas and the received values are different from those received through more detailed expressions. The results of our model simulations of the model for equal intensity of 6 input flows for 2 element copies are summarized and presented graphically in fig. 2 - fig.3. The similar summarized results for different intensity of streams are shown in fig.4 – fig 7. The diagrams show the results of the conducted simulations of the presented GPSS model of centralized 2PL in DDB for different input intervals Tin of transaction coming.

Fig. 2 shows the coefficients of the fixed transactions in same intensities of the incoming streams of global transactions. The greater the intensity, the bigger the coefficients X. The graphics seem smoother when we consider static mode (have a regime close to the stationary regime) (Tn=0,12 min). The graphics in fig. 3 show that when the incoming intensity decreases, probability service of transactions increases, due to the decrease of the probability of locking conflicts.



ig.2. Throughput of the model in one and the ame intensities of the incoming streams

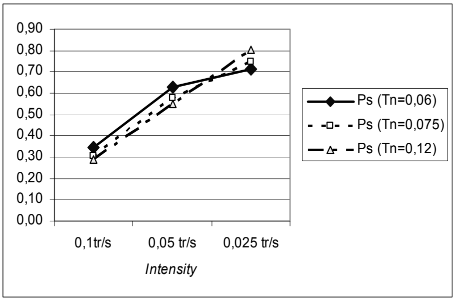
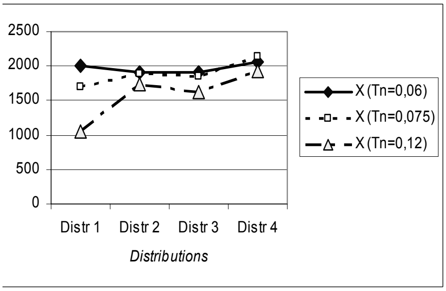


Fig.3. Probability service of the model in one and the same intensities of the incoming streams



ig.4. Throughput of the model in different ntensities of the incoming streams with one nd the same summary intensity 0,6 tr/s

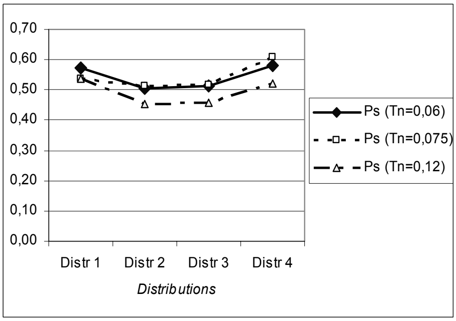
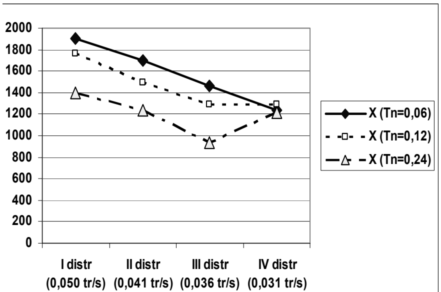


Fig.5. Probability service of the model in different intensities of the incoming streams with one and the same summary intensity 0,6 tr/s



ig.6. Throughput of the model in different ncreasing intensities of the incoming streams

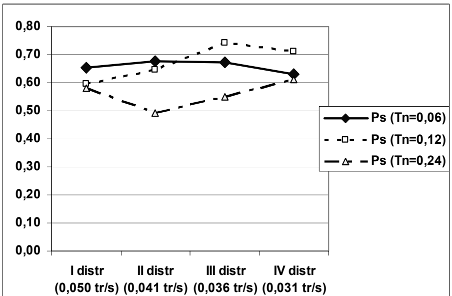


Fig.7. Probability service of the model in different increasing intensities of the incoming streams

The throughput and the probability service depend not only on the intensity of the incoming streams but also on the intensity of the definite incoming stream as one could see in fig. 4 and fig. 5. The intensity of the incoming streams for all of the four distributions is 0,061 transactions per sec and it has different allocation on the site-generators. In some of the distributions the probability for appearing of conflicts for lockings increases and it prolongs the time for global transactions service and decreases the service probability. It is confirmed by the graphics in fig. 5 and fig. 6. Although the summary intensity of the incoming streams from Distribution 1 to Distribution4, in Distribution2 and Distribution3 we notice a decrease of the service probability.

6 Conclusions

We suggest a structural scheme of a modeling algorithm for concurrency control of transactions with priorities in 2PL protocol in DDBMS.

There are developed some fragments of a program GPSS World code for modeling of the distributed transactions service with priority in pessimistic protocols in DDB.

Simulation model allows definition the throughput ability of the distributed system, the average service time of distributed transactions and other parameters on the basis of which the efficiency of the suggested algorithm could be defined.

We continue the gathering of statistics of simulations of the modeling algorithm in different parameters of the system and the coming streams and a comparative analysis of the results in different solutions for the data replication and the concurrency control.

References

1. Abbott, R., Garcia-Molina, H.: Scheduling Real-Time Transactions: A performance Evaluation. In: Proceedngs of the 14th VLDB Conference, (1988)
2. Abbott, R., Garcia-Molina, H.: Scheduling Real-Time Transactions with Disk Resident Data. In: Proceeidngs of the 15th VLDB Conference, (1989).
3. Pavlova, E.: Управление транзакциями в СУБД реального времени. http://meta.math.spbu.ru/~katya/publications/programmirovanie2000
4. Vasileva, S., Milev, A.: Simulation Models of Two-Phase Locking of Distributed transactions. In: International Conference on Computer Systems and Technologies, pp. V.12-1-V.12-6, ACM, New York (2008)
5. Vasileva, S.: 2 Pl Algorithm Model of Distributed Transactions with Primary Copies. In: International Scientific Conference Informatics In the Scientific Knowledge 2008, pp. 237—246, Varna Free University "Chernorizets Hrabar", Varna, Bulgaria (2008)
6. Vasileva, S., Milev, P.: Algorithm modeling Distributed Two-Phase Locking of distributed transactions. In: International Conference Automatics and Informatics'08, Conference Proceedings, pp. VIII-41 – VIII-44, Sofia, Bulgaria (2008)
7. Vasileva, S., Milev, P.: Majority Locking Algorithm model in Distributed Database Systems. In: International Scientific Conference Dedicated to the 105 Anniversary of John Atanasoff and John von Neumann, Shumen, Bulgaria (2008)

VL: Language for Verification of Procedural Programs

Magdalina Todorova
“St. Kliment Ohridski” University of Sofia, Faculty of Mathematics and Informatics
Sofia 1164, Bulgaria
magda@fmi.uni-sofia.bg

Abstract. The article describes a procedural language with axiomatic superstructured semantics and allowing formal verification of procedural and object-oriented programs. The language is built in a HOL theorem prover by depth embedding. Language logic is based on the inference rules of Hoare and uses transformation predicates. The verification of a program (program fragment) regarding given input-output specification is reduced to proving the trueness of a HOL theorem.

Keywords: automated theorem proving, higher order logic, HOL, formal program verification, deep embedding

1 Introduction

The HOL (Higher Order Logic) system [7] is designed to realize interactive proofs of theorems in a logic of a higher order. It has a rich collection of theories and libraries allowing formal proofs of statements from different subject areas.

In order to use those HOL resources, for a number of applications it is required in the logic of HOL to be incorporated different logics including those of programming languages. A survey of embedding programming logics in theorem provers is made in [1]. Known are shallow embedding [1, 2, 5], deep embedding [6] and hybrid embedding [2].

This paper describes a subset of procedural language called VL (Verification Language) and its logic, which are built into the system for proving theorems HOL by deep embedding. In the proposed implementation of the language version 4 of Kananaskis HOL is used [7]. Logic language is based on the Hoare inference rules for verification of programs and uses transformation predicates. By means of the obtained system, programs in the VL language can be verified.

2 The VL language

The language is a procedural one with axiomatic superstructural semantics. For the sake of brevity of this paper, a subset of the language is described. Basic types of data of VL are Integer and Boolean. The integer variables are strings and their values are designated by string preceded by the % symbol. For example "a" and "in" are names of integer variables and % "a" and % "in" are their values. Integer constants are designated by integer numbers preceded by the # symbol. For example #2 and #-6 are integer constants of the VL language.

The VL programs are sequences of procedures and functions composed by the following statements: empty (skip), assignment, sequence, conditional, for loop, for input and output, for addressing a procedure or a function.

0Language grammar

```
arith_expr ::=  % string  | # integer  | arith_expr ADD arith_expr
                | arith_expr SUB arith_expr   | arith_expr MUL arith_expr
                   | arith_expr QUO arith_expr  | arith_expr REM arith_expr
                   | arith_expr POW arith_expr
bool_expr ::= arith_expr EQ arith_expr | arith_expr LS arith_expr
                |  arith_expr LE arith_expr | arith_expr GR arith_expr
                |  arith_expr GE arith_expr  |  bool_expr IMP bool_expr
                |  NOT bool_expr  |  bool_expr AND bool_expr
                |  bool_expr OR bool_expr
command ::=    skip | string :== arith_expr  | command;; command
                |  write arith_expr  | iff bool_expr command command
                | while bool_expr arith_expr bool_expr command
specification ::=  {bool_expr} command {bool_expr} |  [bool_expr]
```

where ADD, SUB, MUL, QUO, REM and POW are arithmetic operators for addition, subtraction, multiplication, integer number division, remainder of integer number division and for raising to power, IMP, NOT, AND и OR mean implication, negation, conjunction and disjunction respectively. The comparison operators are: EQ (for equality), LS (for less), LE (for less or equal), GR (for greater), GE (for greater or equal). Command defines syntaxes of the language statements: *skip* is the empty statement; *string :== arith_expr* is the assignment statement; *command ;; command* is the sequence statement; *write arith_expr* is the output statement; *iff bool_expr command command* is the conditional statement; *while bool_expr arith_expr bool_expr command* is the loop statement, whereas the first two expressions are the invariant and the limiting function of the loop [4], and the last two expressions are the condition and the body of the loop.

The structure {bool_expr} command {bool_expr} is the triad of Hoare and [bool_expr] is the value of bool_expr.

Language logics

Besides the generally accepted procedure semantics, VL logic includes the logic of Hoare [3], described by the following inference rules:

Precondition	1Corollary
[P IMP Q]	{P} skip {Q}
P IMP Q(E←"x")	{P} "x" :== E {Q}
{P} S1 {R} и {R} S2 {Q}	{P} S1 ;; S2 {Q}
[P IMP Q]	{P} write E {Q}
{P AND B} S1 {Q} и {P AND (NOT B)} S2 {Q}	{P} iff B S1 S2 {Q}
[P IMP I] и {I AND B} S {I} и [(I AND (NOT B)) IMP Q] и [(I AND B) IMP (t GR 0)] и [(I AND B) IMP Wp("t1" :== t;; S, t LS t1)]	{P}while I t B S {Q}

where P, Q, R and B are Boolean expressions, E is an arithmetic expression, S, S1 and S2 are arbitrary statements of the VL language, "x" and "t1" are integer variables. The rule of while operator contains the invariant I (Boolean expression) and limiting function t (arithmetic expression) of the loop.

If Prog is a program in the VL language, the verification of Prog regarding P and Q is reduced to proving the trueness of the theorem {P} Prog {Q} [3, 4]. For the purpose, the above described inference rules are applied.

Example. To verify the program fragment, written in the language C++

```
quot = 0;
rem = dividend;
while(divisor <= rem)
{ quot = quot + 1;
  rem = rem - divisor;
}
```

and returning the quotient (quot) and remainder (rem) of the integer numbers division of natural numbers dividend and divisor regarding the precondition

P: $\text{dividend} \geq 0 \wedge \text{divisor} > 0$

and the postcondition

Q: $\text{dividend} = \text{quot} * \text{divisor} + \text{rem} \wedge 0 \leq \text{rem} < \text{divisor}$

with the loop operator are connected the following invariant

I: $\text{dividend} = \text{quot} * \text{divisor} + \text{rem} \wedge 0 \leq \text{rem} \wedge \text{divisor} > 0$

and limiting function

t: rem.

The verification is done by proving the objective:

```
   g `HOR ((#0 LE %"dividend") AND (#0 LS %"divisor"))
              (("quot" :== # 0);;
               ("rem″ :== %"dividend");;
               (while  ((%  "dividend"  EQ  %  "quot"  MUL  %
"divisor"
                                        ADD % "rem") AND
                          (#0  LE  %  "rem")  AND  (#0  LS  %
"divisor"))
                         (% "rem")
                                (% "divisor" LE % "rem")
                         (("quot" :== % "quot" ADD # 1);;
                          ("rem"   :==   %   "rem"   SUB   %
"divisor"))))
               ((% "dividend" EQ % "quot" MUL % "divisor"
ADD % "rem") AND
           (# 0 LE % "rem") AND (% "rem" LS % "divisor"))`;
```

using the HOL system. The HOR function in the objective embeds the inference rules described above.

3 Embedding of VL in HOL

Implemented by embedding the syntaxes and semantics of the VL language into the HOL theorem prover.

03.1 Building in the syntaxes

Implemented by the tools of HOL for definition of data types and functions. The arith_expr type definition is implemented as follows:

```
   val arith_expr =
   Hol_datatype `arith_expr = % of string
               | # of num
               | ADD of arith_expr => arith_expr
               | SUB of arith_expr => arith_expr
               | MUL of arith_expr => arith_expr
               | QUO of arith_expr => arith_expr
               | REM of arith_expr => arith_expr
               | POW of arith_expr => arith_expr`;
```

The arithmetic operators are infixed. Defined by setting the associativeness (left – by means of Infixl or right – by means of Infixr) and their priority. Their priorities are as those of procedure languages. An operation of higher priority is assigned by higher priority number. The structure

set_fixity "POW" (Infixr 99);

assigns the priority of the infix right associative arithmetic operator POW. The remaining arithmetic operators are left associative.

The bool_expr type definition is implemented as follows:

```
val bool_expr =
Hol_datatype `bool_expr =
        EQ  of arith_expr => arith_expr
      | LS  of arith_expr => arith_expr
      | LE  of arith_expr => arith_expr
      | GR  of arith_expr => arith_expr
      | GE  of arith_expr => arith_expr
      | IMP of bool_expr => bool_expr
      | NOT of bool_expr
      | AND of bool_expr => bool_expr
      | OR  of bool_expr => bool_expr`;
```

Embedding of the syntaxes of the language operators can be implemented by means of the definition:

```
val command =
Hol_datatype `command = skip
           | :== of string => arith_expr
      | ;; of command => command
      | write of arith_expr
      | iff of bool_expr => command => command
      | while of bool_expr => arith_expr => bool_expr
   => command`;
```

Operators ":==" and ";;" are infixed and right associative.

13.2 Embedding in the semantics

Realized by embedding the semantics of the arithmetic and Boolean expressions, statements and inference rules. Embedding the expressions and operators is realized using functions Se, Sb and Sc respectively

Se : arith_expr → state → num
Sb : bool_expr → state → {true, false}
Sc : command → state → state

The second parameter of these functions is the state s, at which the expression is calculated or the operator is effected. Function state is of string → num type. In the annex at the end of the paper the definitions of Se, Sb and Sc functions are included.

The definition of the function Sc uses the auxiliary function update. The latter realizes the semantics of the assignment operator by means of a lambda expression of HOL language. Building in of the inference rules is realized using function

HOR : bool_expr → command → bool_expr → {true, false}

The definition of the HOR function is included in the annex and uses the transforming predicate Wp,

Wp : command → bool_expr → bool_expr

The implementation of Wp for the assignment operator is linked with the realization of a substitution of each appearance of the variable V together with the arithmetic expression E in a Boolean expression Q. The latter is done by the function (SUBST_bool Q E V). For Boolean expressions, which are comparisons of arithmetic expressions, substitution is reduced to replacement of each appearance of a variable V together with the expression E in an arithmetic expression A. The latter is done by the function (SUBST_arith A E V).

4 Conclusion

The described language for verification of procedural programs is based on the approach "logical inference" and embedded in a theorem prover HOL uses its resources. The language also includes real structures, commands for loop (for and repeat) and for addressing procedures or functions. A helpful library of theorems, allowing more rapid verification of a large class of procedural programs, is created. The implementation and the experiments with the VL language illustrate the formal verification of small procedural programs and motivate the embedding of tools for a verification of object-oriented programs too. The next task is establishing of a verification environment for imperative and object-oriented programs.

Acknowledgements: The paper is supported by Grant 173/2009 from Sofia University Research Fund.

1References

1. Azurat, A., Prasetya, I.S.W.B.: A Survey on Embedding Programming Logics in a Theorem PVL. UU-CS Utrecht, The Netherlands: Utrecht University: Information and Computing Sciences (2002).
2. Azurat, A. , Prasetya, I.S.W.B.: A preliminary report on xMECH. UU-CS (Ext. rep. 2002-008). Utrecht, The Netherlands: Utrecht University: Information and Computing Sciences (2002).
3. Hoare, C.A.R.: Proof of Correctness of Data Representations, Acta Informatica, Vol. 1, N 4 (1972).
4. Gries, D.: The Science of Programming, Springer-Verlag, Berlin and New York (1981).

5. Prasetya, I.S.W.B., Azurat, A., Vos, T., Leeuwen A., Suhartanto H.: The Theorem PVL Supported Logics for Small Imperative Languages, UU-CS (Int. rep. 2005-046). UU WINFI Informatica en Informatiekunde (2005).
6. Homeier, P.: Proving Programs Correct with the Sunrise Verification, a User's Guide, http://www.cis.upenn.edu/~hol/sunrise/guide.pdf (2005).
7. HOL Documentation http://hol.sourceforge.net/documentation.html (2007).

0Appendix

```
  val Se_def = Define
  `(Se (% v) s = s v) /\
   (Se (# c) s = c) /\
   (Se (m POW n) s = ((Se m s) ** (Se n s))) /\
   (Se (m MUL n) s = ((Se m s) * (Se n s))) /\
   (Se (m QUO n) s = ((Se m s) DIV (Se n s))) /\
   (Se (m REM n) s = ((Se m s) MOD (Se n s))) /\
   (Se (m ADD n) s = ((Se m s) + (Se n s)) ) /\
   (Se (m SUB n) s =  ((Se m s) - (Se n s)))`;

val Sb_def = Define
`(Sb (m EQ n) s = ((Se m s) = (Se n s))) /\
 (Sb (m LS n) s = ((Se m s) < (Se n s))) /\
 (Sb (m LE n) s = ((Se m s) <= (Se n s))) /\
 (Sb (m GR n) s = ((Se m s) > (Se n s))) /\
 (Sb (m GE n) s = ((Se m s) >= (Se n s))) /\
 (Sb (a IMP b) s = ((Sb a s) ==> (Sb b s))) /\
 (Sb (NOT a) s = ~(Sb a s)) /\
 (Sb (a AND b) s = ((Sb a s) /\ (Sb b s))) /\
 (Sb (a OR b) s = ((Sb a s) \/ (Sb b s)))`;

  val Sc_def = Define
  `(Sc skip s =  s) /\
   (Sc (V :== E) s = (update V (Se E s) s)) /\
   (Sc (S1 ;; S2) s = (Sc S2 (Sc S1 s))) /\
   (Sc (write E) s = s) /\
   (Sc (iff B S1 S2) s = if (Sb B s) then (Sc S1 s)
                                     else (Sc S2 s)) /\
   (Sc (while B E B1 C) s =  s)`;

val update = Define
`(update V E s = (\X. if (X = V) then E  else (s X)))`;

val SUBST_arith = Define
```

```
  `(SUBST_arith (% p) E V = (if p = V then E else (%p)))
/\
   (SUBST_arith (#c) E V = (#c)) /\
   (SUBST_arith (m POW n) E V = ((SUBST_arith m E V) POW
                                              (SUBST_arith
n E V))) /\
   (SUBST_arith (m MUL n) E V = ((SUBST_arith m E V) MUL
                                              (SUBST_arith
n E V))) /\
   (SUBST_arith (m QUO n) E V = ((SUBST_arith m E V) QUO
                                              (SUBST_arith
n E V))) /\
   (SUBST_arith (m REM n) E V = ((SUBST_arith m E V) REM
                                              (SUBST_arith
n E V))) /\
   (SUBST_arith (m ADD n) E V = ((SUBST_arith m E V) ADD
                                             (SUBST_arith n
E V))) /\
   (SUBST_arith (m SUB n) E V = ((SUBST_arith m E V) SUB
                                             (SUBST_arith n
E V)))`;

val SUBST_bool = Define
`(SUBST_bool (m EQ n) X V = ((SUBST_arith m X V) EQ
                                         (SUBST_arith n X V))) /\
  (SUBST_bool (m LS n) X V = ((SUBST_arith m X V) LS
                                          (SUBST_arith n X V))) /\
  (SUBST_bool (m LE n) X V = ((SUBST_arith m X V) LE
                                          (SUBST_arith n X V))) /\
  (SUBST_bool (m GR n) X V = ((SUBST_arith m X V) GR
                                          (SUBST_arith n X V))) /\
  (SUBST_bool (m GE n) X V = ((SUBST_arith m X V) GE
                                          (SUBST_arith n X V))) /\
  (SUBST_bool (a IMP b)X V = ((SUBST_bool a X V) IMP
                                          (SUBST_bool b X V))) /\
  (SUBST_bool (NOT a) X V =  NOT(SUBST_bool a X V)) /\
  (SUBST_bool (a AND b) X V = ((SUBST_bool a X V) AND
                                          (SUBST_bool b X V))) /\
  (SUBST_bool (a OR b) X V = ((SUBST_bool a X V) OR
                                         (SUBST_bool b X V)))`;

  val Wp_def = Define
  ` (Wp skip Q = Q) /\
    (Wp (V :== E) Q = (SUBST_bool Q E V)) /\
    (Wp (S1 ;; S2) Q = (Wp S1 (Wp S2 Q))) /\
    (Wp (iff B S1 S2) Q = ((B IMP (Wp S1 Q)) AND (NOT B
IMP (Wp S2 Q)))) /\
    (Wp (while B E B1 S1) Q = B)`;
  val HOR_def = Define
  `(HOR P skip Q = (!s. (Sb P s) ==> (Sb Q s))) /\
 (HOR P (V :== E) Q = (!s. (Sb P s) ==> (Sb (Wp (V :== E)
Q) s))) /\
   (HOR P (S1 ;; S2) Q = ((HOR P S1 (Wp S2 Q)) /\ (HOR
(Wp S2 Q) S2 Q))) /\
   (HOR P (iff B S1 S2) Q = (!s. (Sb P s) ==> (Sb (Wp
(iff B S1 S2) Q) s))) /\
   (HOR P (while B1 E B2 S1) Q = (!s. ((Sb P s) ==> (Sb
B1 s)) /\
                    ((Sb B1 s) /\ (Sb (NOT B2) s) ==> (Sb Q
s)) /\
                    (HOR (B1 AND B2) S1 B1) /\
                     ((Sb B1 s) /\ (Sb B2 s) ==> (((Se E s)
> 0) /\
                                               (Sb (Wp (("t1" :==
E ) ;; S1)
                                                       (E   LS   %
"t1")) s) )))) /\ ...`;
```

Towards a Vision of an Internet of Cultural Things

Mícheál Mac an Airchinnigh
School of Computer Science and Statistics, University of Dublin, Trinity College,
Dublin 2, Ireland
mmaa@cs.tcd.ie

1 Principles

P1. When Tim Berners□Lee (and Robert Cailliau) gave us the World-wide Web [1] he acted exactly according to the first principle of simplicity which in former times was known by Occam's Razor [2] and in modern times as the "Keep it simple stupid" (Kiss) principle. Specifically, he also enunciated exactly how this principle worked in practice: "The trick here… is to make sure that each limited mechanical part of the Web, each application, is within itself composed of simple parts that will never get too powerful." [1] p. 183. We will show how this old principle of simplicity may be applied to semantic web ontology frameworks.

P2. Things are known by their names. Not to name is not to know. My own first name "Mícheál" is Irish Gaelic for "Michael" which is derived from the Hebrew "Who is like God?" [3]. The full name includes the surname (and if needed other naming extensions of family) which is a natural way intended to identify the person. Identification of things (such as cows, paintings, buildings…) in the context of the "Internet Galaxy" [4] is to be done by assigning unique IPv6 names [5].

P3. The child sees (and hears and feels) before it can speak. The eyes are said to be the windows of the soul. The image, the picture, the drawing, the diagram are all perceived directly and immediately. However, such images may not necessarily be understood immediately by the perceiver. A simple example will suffice. The image of the "sky at night" in the Northern Hemisphere was only "really understood" after appropriate interpretation by Copernicus [5] and followers. The new way of seeing (and understanding) is usually named as accepting a "heliocentric cosmology". Geometry (subsequently embracing perspectivity and so on) encodes much of the formality of what is seen and understood culturally. We will demonstrate the different ways of seeing "ontologies" by applying basic geometrical transformations to the 3–dimensional lattice of ontological concepts first published by John Sowa in 2000.

P4. Meaning is in locality. This principle applies to natural language and extends to each and any ontological framework. There can never be a "Universal Language" [6]. In practice, this means translation and interpretation will always be the form. In particular, for example, if one espouses the Getty Terminology [7], one also will have to deal with its relationship to the (international) CIDOC ISO standard [8]. If one has set up the ideal English language ontology for Bulgarian artefacts, one has to negotiate with any other established Bulgarian language ontology for the same set of entities. Put this way, the entire endeavour seems hopeless. But, from the perspective of the machines, acting collaboratively on the Semantic Web, everything is possible.

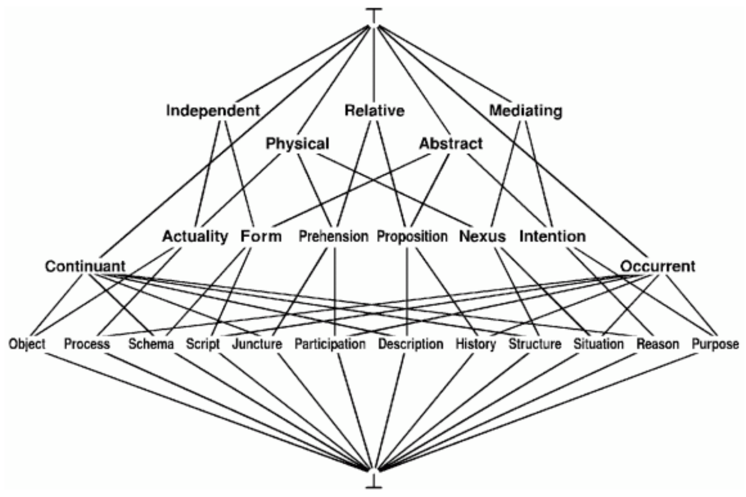


Fig. 9. Data Stream Management System.

In particular, as Berners-Lee put it: *"Inference languages allows computers to explain to each other that two terms that may seem different are in some way the same—a little like an EnglishFrench dictionary. Inference languages will allow computers to convert data from one format to another."* [1] p. 185.

2 The Conceptual Framework

CF1. We have chosen CIDOC [8, 9] to be our top level ontology, for the simple pragmatic reason that it is now an ISO standard [10]. In particular, in any specific application which involves the organization of an Art Exhibition, whether physically or digitally, whether located locally, or abroad, we will assume that the artefacts in question, be they paintings, sculptures, and so on, have been properly classified with respect to CIDOC. For example, (E21 Person) is a subclass of (E20 Biological Object) which in turn is a subclass of (E19 Physical Object). Persons who might be of interest to us are individual painters, sculptors, curators, and so on.

CF2. On the other hand, each one has their own (implicitly learned) ontology by which they negotiate interactions with their world through the language that they use. It is reasonable, therefore, to study the relationship between CIDOC and some other ontological framework. As a comparative top level ontology we have chosen to use the Sowa 12 framework (Figure 1) [11]. This framework consists of exactly 12 concepts, neatly divided equally between Physical (P) and Abstract (A). The same set of 12 concepts may also be divided by time, whether Continuant (C) or Occurrent (O). Finally, there is a third division of the 12 into Independent (I), Relative (R), and Mediating (M). These latter categories may also be labeled 1, 2, 3.

	C Continuant		O Occurrent	
	P Physical	A Abstract	P Physical	A Abstract
I Independent	IPC_object	IAC_process	IPO_state	IAO_script
R Relative	RPC	RAC	RPO	RAO
M Mediating	MPC	MAC	MPO	MAO

Fig. 2. Tabular form for coordinating concepts.

The complete lattice structure of Sowa may be sketched out in a simple tabular form as shown above (Figure 2). Such a form is in reality a coordination scheme. The CIDOC (E21 Person) is quite clearly located under the IPC_object of Sowa's lattice. In the scheme of things sketched out by Berners Lee, the machines (will/ought to) be able to make such an inference automatically.

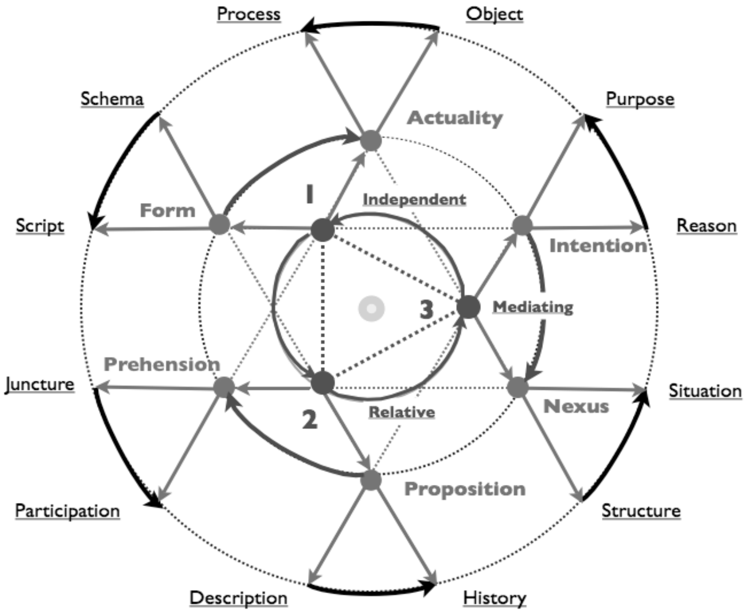


Fig. 3. Sowa 12 rearranged as Wheel of Concepts.

In the setting of the task of constructing the Wheel of Concept out of the 3-D lattice structure of the Ontology of Sowa one naturally begins with the wheel. This small research task (2006) was initially regarded as an elementary geometrical problem. The construction, which took about two weeks, is believed to be original and was presented at the first International Workshop *„Ontology Based Modelling in the Humanities"*, University of Hamburg, on Friday 7th April 2006 [12]. We present a brief outline of the conceptual and historical basis of the construction. The number 12 is significant in Mathematics. There are many ways to point out that significance. Culturally, let us mention the 12 signs of the Zodiac, the 12 Apostles in the Christian tradition, the 12 stars on the EU symbol, the 12 months of the year, and the 2 x 12 hours of the day, and of course the 12 tone equal temperament of (modern) western music, often associated with Bach [13]. The conceptual mathematician will immediately point out that 12 is 2x6 and 6 is the first perfect number. Further, the 6 is 2x3 and 3 is the number of "God". Geometrically, the 3 is denoted by the equilateral triangle. And a pair of equilateral triangles will give the 6 pointed star.

The cycle (permutation), circle (geometry), and ring (algebra) all have particular mathematical meaning. The term **wheel** seemed to be appropriate for similar special mathematical investiture. Hence, there came into being the mathematical "wheel of concepts." This naming, and investiture with meaning, of the wheel pointed to the richness of the concept, For example, the "Wheel of Life" is wellgrounded in many cultures. See, for example, the Bhavacakra [27] which has 6 sections and 12 links of causality. There is also something democratic about the well (which re grounds the abstraction of cycle and circle and ring). Bach also comes to mind with the introduction of the 12 semitone scale [13]. One might also have imagined, even presumed, that the Wheel of Fortune [28] in Carl Orff's opera "Carmina Burana" was also a 12 spoke wheel. It was not. Many of the associated images are of 8 spoke wheels. Had it been universally 12-spoked, then it would have provided a nice point of "musical" closure before presenting the construction below.

In order to comprehend the construction, let us focus on the most significant feature of the diagram which is the occurrence of the directed arrow • → • . The 12 concepts are given in counterclockwise order from the top, beginning with **Object**, **Process**, **Schema**, **Script**… These are the **4** Independent concepts. The order chosen, and the starting position, follow the usual clock order, but in reverse. A natural first step in contructing the wheel, is to build on the schema of the clock itself.

The greatest problem then faced was the filling in of the interior. In other words, how was one to inject the bases into the ring? The notes recorded in 2006 suggest that this conceptual world of "philosophical soup" could be divided very naturally in the **Physical** and the **Abstract** [12]. This provides a classical orthogonal basis for a 2-dimensional vector space. Two into twelve gives six. The concept of six provoked the idea of two intersecting equilateral triangles which suggested the classical six point star. Although the diagram of the wheel is flat, the actual structure is not. To try to capture the 3-D nature, the center is marked by two disks of complementary colors: yellow for **thing** and blue for **not-thing**, also known as **top** and **bottom**. In other words, the original 3-D lattice of Sowa is being "circularized".

Although this stage of the re-construction is remarkably pretty, it does not yet give the right picture of the top level concepts of the ontology. The next stage of the construction (just as in a real wheel) is the binding with cycles.

At the heart of the Sowa construction of the Top-level categories is the **core of 3** in which everything can be described. This core may be denoted by Oneness (**1**), Twoness (**2**), and Threeness (**3**) or Independent (**I**), Relative (**R**), and Mediating (**M**) concepts, respectively. The core is a mathematical vector space with basis of dimension 3. Where will it fit in to the diagram already constructed? Clearly, it must be pictured as an equilateral triangle within the cycles at the very center. Knowing the first 4 Independent concepts, already introduced above, and given the existing architectural geometry, everything is already completely constrained. Hence it is only a matter of fitting it into its proper place at the right size and orientation. To achieve this goal, the 12-pointed star is first deleted. Then it is clear how the 1 (Independent) vertex should be located, and the rest follows.

Now we have constructed a vector space of dimension 6. Using the basis vectors <I, R, M> and <P, A>, we can construct the 6 intermediary concepts <IP, IA, RP, RA, MP, MA>. Sowa gives these the names **Actuality** (IP), **Form** (IA), **Prehension** (RP), **Proposition** (RA), **Nexus** (MP) and **Intention** (MA) [11].

To introduce the final step in the construction by introducing the vector basis of dimension 2 dealing with "time". The names of the concepts used are **Continuant** (C) and **Occurrent** (O). An object is a continuant in the sense that it does not change over (the) time (interval of the ontological framework in question). A process is really just like an object except that it is continually changing. Or, to put it differently, everything is really a process. It is the time interval which determines how we see it. For example, both the human body and glass are processes, i.e., they are in the state of continuous change. Everyone can see how the body changes over time. It takes a long time (longer than a human lifetime) to see that glass is a liquid. Diamond might be a better example. Recalling that the revolution of the Wheel of Concepts has been arbitrarily chosen to be counterclockwise then we obtain the final geometrical construction. This completes the transformation of Sowa's lattice.

Referring back to the original cyclic diagram one notices that there are directional arrows yet to be accounted for. For example, there is a counterclockwise directed arrow from IPC (Object) to IPO (Process). Such an arrow captures the idea of change of the continuant over time. In other words, in the bigger temporal picture, every continuant becomes a process. This discovery of the existence of such a directed arrow implied that the diagram had intrinsic mathematical properties, yet to be revealed. One was dealing with objects and arrows. Hence, it was to be expected that the diagram might itself be mathematically categorical.

In the final diagram illustrating the categorisation of the Wheel of Concepts we introduce the classical computing theory of a formal language to label computational paths by words over an alphabet. We now can see that the Sowa 12 Top level concepts can be organised as 3 binary trees with a common root „**THING**". The roots of each tree are respectively labelled I, R, M. Let us focus on the Independent binary tree which is the backbone of everythings. If we go one step to the right, we reach IP, or if we go one step to the left we reach IA. In other words, Actuality (IP) is given by the word I**R**, where **R** here denotes right turn, and Form (IA) is given by the word (I**L**), where **L** denotes **left** turn. Thus Object is given by the word IRR and Process by the word IRL. Finally, the arrow Q which maps the Continuant to the Occurrent may be regarded as a reduction operator on words, giving IRR**Q** = IRL. This is nothing more that the standard commuting diagram on triangles, where R**Q** = L.

Now we turn to the inner circle where we have a map operator **K** that takes Form (IA) to Actuality (IP). This operator encodes the concept that Form characterises Object [11] and in general, Abstract categories characterise Physical categories. Hence, using this new operator, we can map Form to Object in terms of an equation on words: IL**K** = IR, another commuting diagram on triangles. This permits us to see (i.e., deduce categorically) that Script characterises Process and Schema characterises Object (by following the arrows IAO**K** = IPO and IAC**K** = IPC).

CF3. To complete the (local) Conceptual Framework, recourse is had to Roget's thesaurus [14-16}. Although the Roget thesaurus dates from 1852, the underlying philosophical framework is grounded on the work of Bishop John Wilkins [17], an excellent account of which is provided by a chapter of Umberto Eco's book on "The Search for the Perfect Language" [6]. In particular, it is noteworthy that Eco concludes (p. 259) that perhaps *"the defect in Wilkins' system [was] its prophetic virtue? What if we treated Wilkins as if he were obscurely groping towards a notion for which we have only recently invented a name — hypertext?"* If indeed Eco's intuition is right, then Roget continues that trend of groping towards hypertext. Many others subsequently contributed to and developed the concept of hypertext that we take for granted today. Of all those, it would be Tim Berners-Lee who would get the ultimate credit through the launching of the World-wide Web.

Pragmatically, everything is encoded in the Semantic Web Ontology Language, restricted to the description logic level. For this purpose we use Protégé-OWL, the "Ontology Editor for the Semantic Web" [18].

4 Populating

In 2008 the EU Commission put out a call for response to its staff document "Early Challenges regarding the "Internet of Things" [19]. Among the published responses, the author contributed a 10-page document "Towards a vision of an Internet of Cultural Things" [20] with the intention of highlighting the potential of the technology to revolutionize the management and use of (digital) cultural heritage. From a pragmatic perspective it is the people who will determine whether or not the digital culture will become an authentic extension of the existing cultural heritage. How will the people determine the development?

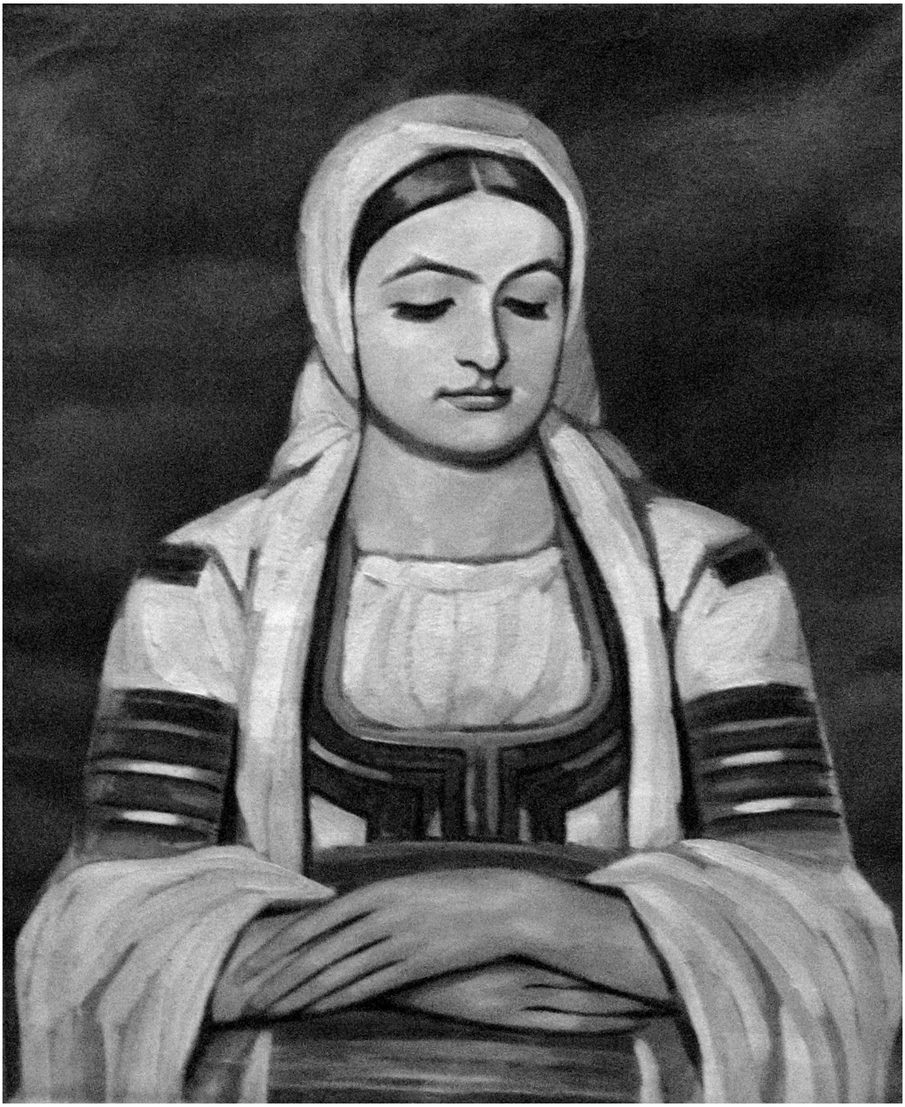
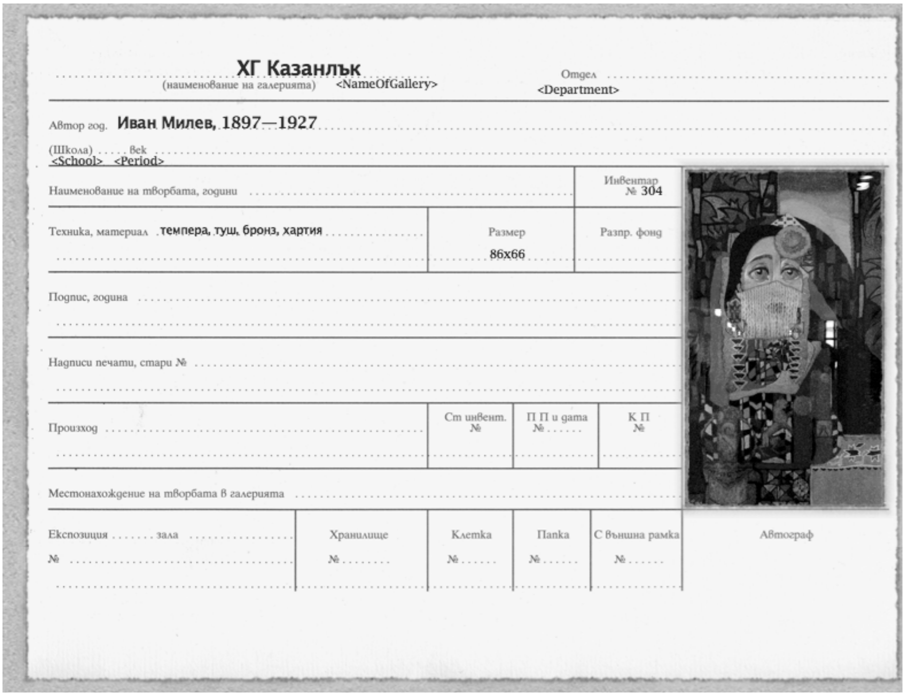


Fig. 4. Majstora's Kjustendil Woman in Smolyan Art Gallery

Example 1: Paintings and Passports: It is conventional in Bulgaria (and doubtless in other countries also) that each painting in an Art Gallery should have a complete dossier, describing in precise detail everything known about it: provenance, fund by which acquired, etc. However, for practical purposes, each painting should have its own passport that records the most essential information that might be needed, in case of preparing an Exhibition Catalog, for instance. Such a passport might travel with the painting.



ХГ Казанлък
(наименование на галерията) <NameOfGallery>

Отдел
<Department>

Автор год. Иван Милев, 1897—1927

(Школа) ..... век
<School> <Period>

Наименование на творбата, години

Инвентар
№ 304

Техника, материал темпера, туш, бронз, хартия

Размер
86x66

Разпр. фонд

Подпис, година

Надписи печати, стари №

Произход

Ст инвент.
№

П П и дата
№ ......

К П
№

Местонахождение на творбата в галерията

Експозиция .... зала
№

Хранилище
№ ....

Клетка
№ ......

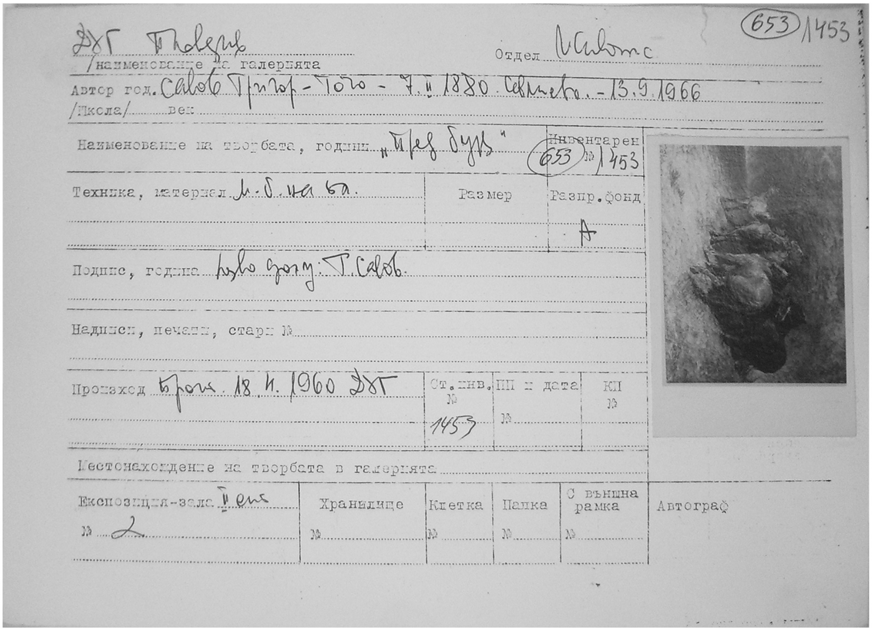
Папка
№ ......

С външна рамка
№ ......

Автограф

Fig. 5. Working sketch of actual passport for famous Ahinora painting of Ivan Milev.

In the illustration shown, it can be seen that digital passports of paintings will be constructed to conform with the “traditional form” in a backwards compatibility strategy and will be available in multi-lingual forms: that of the home country (here shown in Bulgarian) and that of the target exhibition country (say France or United States or…). Traditionally, passports of paintings have been filled in by hand.



ДХГ Пловдив
/наименование на галерията

Отдел Живопис

(653) 1453

Автор год. Савов Григор - Гого - 7.II 1880 Севлиево - 13.9.1966
/Школа/ век

Наименование на творбата, години „Пред буря"

Инвентарен № (653) 1453

Техника, материал м.б. на ба.

Размер

Разпр. фонд А

Подпис, година долу дясно: Г. Савов.

Надписи, печати, стари №

Произход Откуп. 18.4.1960 ДХГ

Ст. инв. № 1453

ПП и дата №

КП №

Местонахождение на творбата в галерията

Експозиция-зала II ет. № 2	Хранилище №	Клетка №	Папка №	С външна рамка №	Автограф

Fig. 6. Copy of a traditional passport of a painting, courtesy of Plovdiv Art Gallery.

The design of such passports (and the associated dossiers) for art works in the digital age is urgent. To achieve it requires enormous collaborative work by those fluent in Bulgarian and "XML derivatives". To suggest how this might be achieved in the Internet Galaxy, the following two examples might be considered paradigmatic.

Example 2: Galaxy Zoo [21, 22]: users tag images of galaxies from the comfort of their own homes.

Example 3: Google Earth [23]: users tag the earth with names (in their own languages) and images (usually digital photographs). The latter are often misplaced, in the sense of not being precise with respect to location. Moreover, there is usually no indication as to the direction and orientation of the photographer who points the camera to shoot the scene.

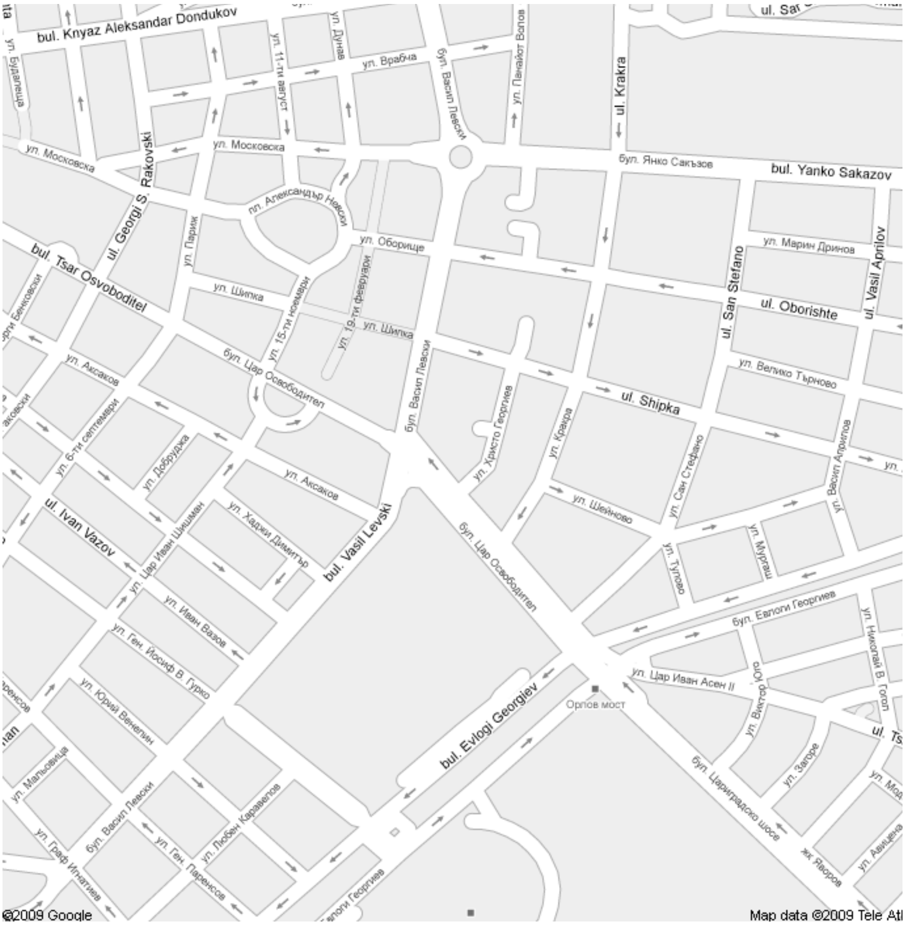


Fig. 7. Sofia University as seen by Google Maps after being Google Earthed.

In Google Earth one gets to a place by "flying", recalling (perhaps inadvertently) the manner of travel in Second Life. For example, to get to Sofia, Bulgaria is easy. To get to the University of Sofia requires a little more effort. That is to say, by "flying" many possibilities are offered. One needs to know something of the local geography and the academic world. In this case, it turns out that the University of Reading (UK) has already put down an accurate place marker entitled "Sofia University". I know that this is the right place. I also know that this is where the Faculty of Philosophy is located and that the Faculty of Mathematics lies elsewhere. From Google Earth I can go directly to Google Maps and print out the result (Figure 7). That is to say I can show you exactly what I have been talking about via a specific image. I could also have included a video clip of how I achieved this task. Finally, it is worth noting that it is possible to associate a Wikipedia article with a particular place. Instructions on how to do this are available at Frank Taylor's Google Earth Blog [24].

References

1. Berners-Lee, T. and M. Fischetti, Weaving the Web: the original design and ultimate destiny of the World Wide Web by its inventor. 1st pbk. ed. 2000, New York: HarperCollins Publishers. ix, 246 p.
2. W en. Occam's razor. [cited 2009 Mar 12]; Available from: http://en.wikipedia.org/wiki/Occam's_razor.
3. W en. [cited 2009 Mar 12]; Available from: http://en.wikipedia.org/wiki/Michael.
4. Castells, M., The Internet galaxy : reflections on the Internet, business, and society. 2001, Oxford ; New York: Oxford University Press. xi, 292 p.
5. W en. IPv6. [cited 2008 Nov 14]; Available from: http://en.wikipedia.org/wiki/IPv6.
6. Eco, U., The search for the perfect language. The making of Europe. 1995, Oxford, UK ; Cambridge, Mass., USA: Blackwell. x, 385 p.
7. The Getty Information Institute. Guidelines for Forming Language Equivalents: A Model Based on the Art & Architecture Thesaurus. [cited 2006 Dec 2]; Available from: http://www.chin.gc.ca/Resources/Publications/Guidelines/English/.
8. Nick Crofts, et al. CIDOC Conceptual Reference Model, version 4.2.2. 2008 [cited 2008 Jan 27]; Available from: http://cidoc.ics.forth.gr/.
9. W en. CIDOC Conceptual Reference Model. [cited 2009 Mar 6]; Available from: http://en.wikipedia.org/wiki/CIDOC_Conceptual_Reference_Model.
10. ISO 21127:2006. Information and documentation A reference ontology for the interchange of cultural heritage information. 2006 [cited 2009 Mar 6]; Available from: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=34424.
11. Sowa, J.F., Knowledge representation: logical, philosophical, and computational foundations. 2000, Pacific Grove, CA ; London: Brooks/Cole. xiv, 594.
12. Mac an Airchinnigh, M. Original Diagram for the Sowa 12 (http://web.mac.com/micheal1/Михал_Орела/Blog/Entries/2006/6/7_Сряда_%3A_Original_Diagram_for_the_Sowa_12.html). 2006 [cited 2008 June 14]; Available from: http://web.mac.com/micheal1/.
13. W en. Twelvetone Equal Temperament. 2008-06-13 [cited 2008 June 13]; Available from: http://en.wikipedia.org/wiki/Twelvetone_equal_temperament.
14. Roget, P.M. Roget's Thesaurus (1911). [cited 2009 Mar 8]; Available from: http://machaut.uchicago.edu/rogets.
15. W en. Roget's Thesaurus. [cited 2009 Mar 8]; Available from: http://en.wikipedia.org/wiki/Roget's_Thesaurus.
16. Roget, P.M., Roget's Thesaurus, ed. S.M. Lloyd. [1852] 1962, 1982, Burnt Mill, Harlow, Essex: Longman Group Limited.
17. W en. John Wilkins. [cited 2009 Mar 9]; Available from: http://en.wikipedia.org/wiki/John_Wilkins.
18. Protégé OWL. Ontology Editor for the Semantic Web. [cited 2009 March 6]; Available from: http://protege.stanford.edu/plugins/owl/download.html.
19. Commission Staff. Early Challenges regarding the "Internet of Things". [cited 2008 Nov 14]; Available from: http://ec.europa.eu/yourvoice/ipm/forms/dispatch?form=IOTconsultation.
20. Mac an Airchinnigh, M. Towards a vision of an Internet of Cultural Things. [Position Paper] 2009; Available from: http://ec.europa.eu/information_society/policy/rfid/library/index_en.htm#iotcons.
21. Charman-Anderson, S. A galazy of helpful people. 2009 [cited 2009 Mar9]; Available from: http://www.guardian.co.uk/technology/2009/jan/15/internetastronomy.
22. Lintott, C. Galaxy Zoo. [cited 2009 Mar 9]; Available from: https://galaxyzoo.org/.

23. Technophile. Google Earth. 2009 [cited 2009 Mar 9]; Available from: http://www.guardian.co.uk/technology/2009/feb/12/google-earthtechnophile.
24. Taylor, F. Google Earth Blog. [cited 2009 Mar 9]; Available from: http://www.gearthblog.com/blog/archives/2006/12/multilingual_wikiped.html.
25. Breitman, K.K., M.A. Casanova, and W. Truszkowski, Semantic Web; Concepts, Technologies and Applications. 2007: Springer.
26. Sowa, J.F. Top Level Categories. 2001 [cited 2006 January 10]; Available from: http://www.jfsowa.com/ontology/toplevel.htm.
27. W en. Wheel of Life. 2008-06-13 [cited 2008 June 13]; Available from: http://en wikipedia.org/wiki/Wheel_of_life.
28. W en. The Wheel of Fortune. 2008-06-14 [cited 2008 June 14]; Available from: http://en.wikipedia.org/wiki/The_Wheel_of_Fortune.

Grid INFN Virtual Laboratory for Dissemination Activities (GILDA)

Roberto Barbera[1,2], Leandro N. Ciuffo[1], Emidio Giorgio[1]
Antonio Calanducci[1], Valeria Ardizzone[1]
[1] Istituto Nazionale di Fisica Nucleare (INFN), Via S. Sofia 64,
95123 Catania, Italy
[2] Dipartimento di Fisica e Astronomia dell'Università di Catania, Via S. Sofia 64,
95123 Catania, Italy
{roberto.barbera, leandro.ciuffo, emidio.giorgio, tony.calanducci, valeria.ardizzone}@ct.infn.it

Abstract. The Grid INFN virtual Laboratory for Dissemination Activities (GILDA) is a fully fledged Grid test-bed devoted to training and outreach activities. Open to anyone who wants to have its first hands-on experience with grid systems, GILDA has been adopted as the official training infrastructure by many Grid projects all around the world. All services, tools and materials produced in the past tutorials can be freely used by anyone who wants to learn and teach Grid technology. Additionally, through a set of applications ported on its Grid Infrastructure, developers can identify components and learn by examples how to "gridify" their applications. This work presents the main features of such training Infrastructure.

Keywords: Grid computing, Training, Virtual Laboratory, gLite, e-infrastructure.

1 Introduction

Launched in 2004 by the Italian National Institute for Nuclear Physics (INFN), GILDA (the Grid INFN virtual Laboratory for Dissemination Activities) is a fully fledged Grid test-bed devoted to dissemination activities. This infrastructure is open to anyone who wants to have its first hands-on experience with Grid systems. Actually, GILDA can be an important tool for at least three main categories of users:

1. Grid newcomers – people willing to start learning how to use a grid infrastructure;
2. Grid application developers – Just like "webifying" applications to run on a web browser, grid users need to "gridify" their applications to run on a Grid. Through a set of applications ported on GILDA, developers can identify components and learn by examples how to "gridify" their own applications;
3. Tutors – GILDA has been developed keeping training and education in mind. Thus, all services, tools and materials produced in the past tutorials can be freely used by anyone who wants to learn or teach Grid technology.

Indeed, GILDA has been adopted as the official training platform by several former and current Grid projects, such as EGEE/EGEE-II/EGEE-III [1], EELA/EELA-2 [2], EU-IndiaGRID [3], EUMEDGRID [4], EUChinaGRID [5], ICEAGE [6], OMII-EU [7], BEinGRID [8] and many others, becoming a "de facto" standard training-Infrastructure (referred hereafter as t-infrastructure) in Europe and in several other parts of the world for dissemination of Grid Computing technology.

GILDA objectives can be summarized as follows: (i) to raise awareness of Grid Computing benefits; (ii) to provide customized formats for dissemination events, according to the skills of attendants; (iii) to facilitate appropriate free on-line content and services for training purposes; and (iv) to encourage the use of a complete t-infrastructure by new communities.

This article aims at presenting an overview of GILDA facilities as well as to invite the reader to try such t-infrastructure.

2 Background

Computational and storage limitations are key issues for organizations that depend on computation-intensive applications. Such organizations are frequently affected by market pressures to reduce deployment time and maintenance costs. Hence, they may be looking for ways to improve the effectiveness of their infrastructure or their business processes through transformation, or seeking opportunities for innovation that will benefit the business. Grid is not the answer by itself, but in many of these cases, it can certainly play an important role, allowing immediate productivity and benefits and giving more choices and control on how to purchase and leverage IT power for competitive advantage. The Grid vision is to expand parallel and distributed computation, providing a virtualization of heterogeneous compute and data resources, supporting security policy based resource allocation and prioritization. The Grid is ideal for any applications requiring excellent performance and scalability for their compute-intensive processes (e.g., Monte Carlo simulation, engineering CAD simulations, protein modeling, 3D rendering, computational archaeology investigations, etc.).

In this context, the GILDA Project was born with the aim of offering a one-of-a-kind service for those interested in testing the Grid, using gLite [9] and the EGEE infrastructure with their own systems. GILDA offers either basic experiences through the “Try the Grid” [10] walkthrough in minutes or intensive and in-depth training by helping users willing to develop applications to be ported into a Grid environment**.**

3 Dissemination Tools

The main objectives of GILDA activity around the world are to encourage and help new and existing communities to support them for improvement or migration of their applications to Grid infrastructures, to accelerate the adoption of Grid technologies, and to increase the satisfaction of those currently using the Grid services through the communities’ feedback. Training activities are a key component of the knowledge dissemination process, ensuring that all users fully understand the characteristics of the offered Grid services and that they have enough expertise to properly use the available Grid infrastructure. In order to achieve the main objectives, several dissemination instruments are used. A brief description of these instruments is presented below.

3.1 Tutorials for Applications Developers

Porting an application into a Grid environment has never been an easy task. In order to enable application developers to get used to the main Grid functionalities, an advanced tutorial has been created fully dedicated to teach them how to “gridify” their applications. Such training events are the perfect scenario to put application experts in tight collaboration with Grid experts.

Hence, the GILDA Team has acquired a good experience in transferring knowledge and know-how to new communities by helping them to integrate their applications into several Grid projects’ infrastructures.

3.2 Tutorials for Sysadmin

Tutorials for grid system administrators are organized by the GILDA Team to meet the needs of computer centers interested in joining the GILDA test-bed or other Grid infrastructures. It is also worth mentioning that it is possible to have your own GILDA-like stand-alone Grid installed in your institution. In fact, it is a common practice adopted to support the organization of long training events, such as Grid Schools or graduation courses.

Usually, for a novice user, installing a new Grid service and configuring it properly are not straightforward tasks. In that scenario, a step-by-step guide is desired to help user during the troubleshooting stage. The appropriate documentation on site installation can be found at [20].

The sysadmin tutorials are divided into three parts: a theoretical part, a practical one and the hands-on section. In each presentation, the GILDA tutors use slides to show the main steps necessary to install and configure a Grid element. Then, in the hands-on section all participants try to install a Grid element by themselves, assisted by tutors.

Training material is composed of multimedia slides, a Wiki website and a set of "Virtual Services", detailed below.

The most important stage during the preparation of a tutorial is to set up the machines used for the grid node installation: to make it easy, a virtualization technique such as VMware® is adopted in all organized tutorials. In this way, all participants have their own Virtual Machines on which they can work with minimum efforts required from the system administrators.

3.3 GILDA Wiki

The GILDA Wiki [11] has been created mainly with the purpose of documenting and organizing the huge amount of training material produced so far. Its contents are freely available on the Web for every interested user.

Moreover, this site is not a simple on-line documentation repository, but an important collaborative tool used by the whole project team. Thus, all registered user can contribute feeding the site with useful additional training material.

The Wiki site is an information source for three target audiences: users, site administrators and application developers. Regarding the users' content, the available material is subdivided into three different complexity levels (Basic, Medium and Advanced), while the site administrators area consists in step-by-step procedures to install and configure Grid services. Finally, the developers section presents the middleware functionalities that can be integrated into the source code using the API.

3.4 Support System

Through a ticket-based support system [12], users can send their questions to the GILDA Team and get customized answers to their issues. The GILDA support system is an important communication tool aimed at helping anyone facing problems on using or installing a GILDA site.

3.5 Virtual Services

Through the use of the virtualization technique, it is possible to carry out all Grid elements. This instrument has many benefits, such as increasing services' portability and reducing both the time to put a site in operation and the number of real machines required. The complete list of available virtual services is available at [13].

4 The GILDA Test-Bed

The GILDA test-bed is maintained on a "best-effort" basis. The computational resources usually comes from institutions located in Europe, Asia or Latin America. Therefore new grid sites, using heterogeneous hardware, are frequently joining the t-Infrastructure, just like a "real world" Grid environment.

The test-bed itself is made up of all the components of a larger real Grid infrastructure, including services, resources and monitoring tools. To allow the use of the test-bed, it also features a Virtual Organization (VO) and a real Certification Authority (CA) that grants two-week certificates for the test use of the GILDA infrastructure. In addition, it runs the latest production (and stable) version of the gLite middleware [9] developed in the context of the EGEE project.

In order to ease the access to its infrastructure, GILDA also provides a grid portal called GENIUS [15], where different usages are supplied, such as basic content for novice users and full featured ones for more in depth tutorials and demonstrations. At a more advanced level, GILDA offers a one-of-a-kind service for those interested in testing the Grid and gLite with their own software, offering a more intensive and in-depth introduction to the Grid. In a couple of weeks, an user can be trained in Grid usage, his/her software can be modified in order to run on the Grid environment and finally, a GENIUS service can be created to increase the visibility of the work that has been done.

5 Virtuous Cycle

By its nature, GILDA is one of the key enablers of the "virtuous cycle" to attract and support new communities. Its workflow, depicted in Figure 2, is described a below:

1. Driven either by the dissemination events or scientific papers where GILDA is cited, a novice user can get the feeling of what Grid Computing is and which kind of applications can run on a Grid infrastructure by exploring the GILDA web portal or using the Grid Demonstrator interface [14];
2. Following up the dissemination activities, institutions usually request the organization of Grid training events based on the GILDA t-infrastructure. An interested user, participating in such tutorial or induction course, can go through all the mandatory procedure of the request of a personal digital certificate and subscription to a Virtual Organization and then use the Grid Tutor machine [15];
3. After participating in a training event or even doing self-training by following the GILDA wiki pages, many users try to port their own applications on the test-bed. In a smaller scale, these users will face all problems of interfacing the Grid services available before entering in a huge production quality e-Science infrastructure;
4. Various applications from different communities ported on GILDA can be either deployed on large e-Infrastructures or incorporated into the Grid Demonstrator, enriching the portfolio of examples that can be demonstrated to new people.

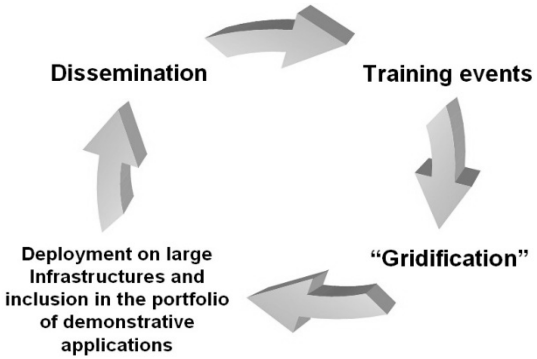

Dissemination
Training events
"Gridification"
Deployment on large Infrastructures and inclusion in the portfolio of demonstrative applications

Fig. 2. "Virtuous cycle" of applications on GILDA

6 Applications

The Grid environment requires many new skills both for scientists that need to learn how to work on it and for application developers that have to learn how to write and optimize codes to properly interact with the available Grid services and computational resources. As a consequence, disciplinary support is also essential to carry out Grid knowledge management and Grid education, both generic and towards specific application domains. In GILDA a set of "Case Studies" of integrated applications has been created, so that scientist or software developers can explore and compare different approaches for the integration process of their application on a Grid infrastructure.

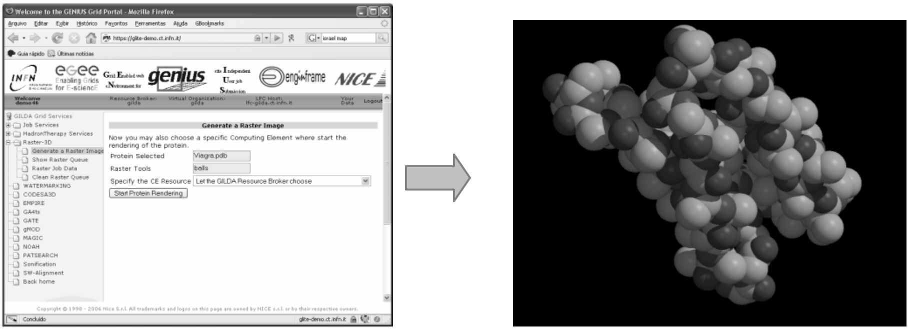
Currently, GILDA applications portfolio covers a large range of different research domains. A short list of applications successfully ported into GILDA test-bed is presented below.

6.1 GATE

Gate is a C++ platform based on the Monte Carlo Geant4 toolkit [18]. It has been designed to model nuclear medicine applications, such as PET (Positron Emission Tomography) and SPECT (Single Photon Emission Computed Tomography) in the context of the OpenGATE collaboration [19]. Its functionalities combined to its ease-of-use make this platform also impressive for radiotherapy and brachytherapy treatment planning.

6.2 RASTER 3D

Raster3D is a set of tools for generating high quality raster images of proteins and/or other molecules. It uses an efficient software Z-buffer algorithm which is independent of any graphics hardware. Figure 3 shows the input interface and an exampled of generated output from this application.



Generate a Raster Image

Now you may also choose a specific Computing Element where start the rendering of the protein.

Protein Selected: Viagra.pdb
Raster Tools: balls
Specify the CE Resource: Let the GILDA Resource Broker choose

Copyright © 1998 - 2006 Nice S.r.l. All trademarks and logos on this page are owned by NICE s.r.l. or by their respective owners.

Fig. 3. RASTER 3D – input interface / output visualization scheme

6.3 Volcano Sonification

Current knowledge of volcanic eruptions does not yet allow scientists to predict future eruptions. This application represents an attempt to put the scientific community one step closer to the prediction asset by means of the sonification of volcano seismograms. Thus, the translation of the patterns of Mount Etna (Italy) and Mount Tungurahua (Ecuador) volcanic behaviors into sound waves has been carried.

Data sonification is currently used in several fields and for different purposes: science and engineering, education and training. It acts mainly as data analysis and interpretation tool. Figure 4 illustrates its working scheme.

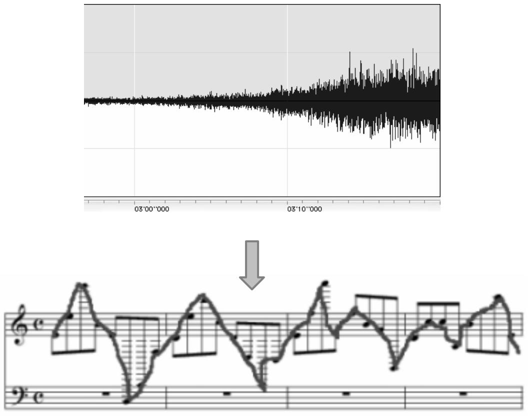


Fig. 4. Volcano Sonification – input / output scheme

6.4 Hadron Therapy

HadronTherapy simulates the beam line and particles detectors used in the proton-therapy facility CATANA (Centro di AdroTerapia e Applicazioni Nucleari Avanzate), operational at INFN-LNS. Here, a 62 MeV (62 million of electron volts) proton beam, accelerated by a superconducting cyclotron, is used for the treatment of some kind of eye cancer.

7 Forthcoming Features

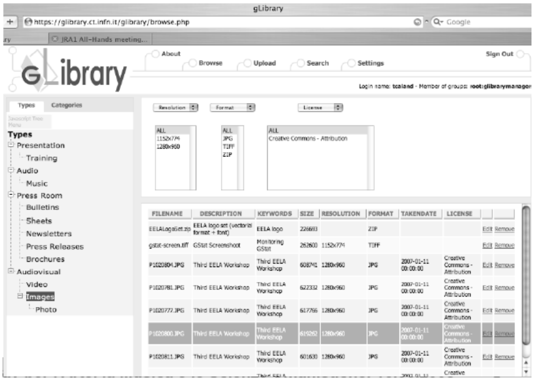
The t-infrastructure provided by GILDA will be continuously used by many EU funded projects, since several project proposals submitted to the 7th Framework Programme calls expressed their intention to use GILDA facilities as part of their training program.

The GILDA hardware expansion plan foreseen the integration of new 64-bits worker nodes by the end of 2008 (up to now, only 32-bits nodes are available). This will lead the availability of new libraries often requested by application developers and the general industry community such as MPI-2.

Additionally, the growing number of users has encouraged the GILDA team to recently announce the incorporation of some new features. These features are:

7.1 gLibrary

gLibrary [21], an extensible, robust, secure and easy-to-use system to handle digital libraries in a Grid infrastructure. It offers an intuitive Web interface to browse the available entries, and thanks to its powerful "iTunes-like" searching capabilities based on attribute filters, finding a library asset is just a matter of seconds. gLibrary is a flexible system that can be used for different purposes and for different communities that can easily adopt this framework to build their own digital libraries defining types and categories according to their needs.



Login name: tcaland - Member of groups: root:glibrarymanager

Resolution	Format	License
ALL	ALL	ALL
1152x774	JPG	Creative Commons - Attribution
1280x960	TIFF	
	ZIP	

FILENAME	DESCRIPTION	KEYWORDS	SIZE	RESOLUTION	FORMAT	TAKENDATE	LICENSE		
EELALogoSet.zip	EELA logo set (vectorial format + font)	EELA logo	226693		ZIP			Edit	Remove
gstat-screen.tiff	GStat Screenshoot	Monitoring GStat	262600	1152x774	TIFF			Edit	Remove
P1020804.JPG	Third EELA Workshop	Third EELA Workshop	608741	1280x960	JPG	2007-01-11 00:00:00	Creative Commons - Attribution	Edit	Remove
P1020781.JPG	Third EELA Workshop	Third EELA Workshop	622332	1280x960	JPG	2007-01-11 00:00:00	Creative Commons - Attribution	Edit	Remove
P1020777.JPG	Third EELA Workshop	Third EELA Workshop	617766	1280x960	JPG	2007-01-11 00:00:00	Creative Commons - Attribution	Edit	Remove
P1020800.JPG	Third EELA Workshop	Third EELA Workshop	619262	1280x960	JPG	2007-01-11 00:00:00	Creative Commons - Attribution	Edit	Remove
P1020811.JPG	Third EELA Workshop	Third EELA Workshop	601630	1280x960	JPG	2007-01-11 00:00:00	Creative Commons - Attribution	Edit	Remove

Fig. 5. gLibrary: library browsing with attribute filters

7.2 Secure Storage Service

This service was carried out by UNICO S.r.l. [22] in close collaboration with INFN [23] in the context of the TriGrid VL Project [24] . This service provides the users with a set of tools for storing confidential data (e.g., medical or financial data) in a secure way and in an encrypted format on the Grid storage elements. The data stored in this way are accessible and readable by authorized users only. Moreover, it solves the insider abuse problem preventing also the administrators of the storage elements from accessing the confidential data in a clear format.

7.3 GRelC

The Grid Database Access Service (GRelC) [26] aims to provide a set of advanced data grid service to transparently, efficiently and securely manage databases in a Grid environment [27]. This framework, developed by University of Lecce – Italy and SPACI Consortium (The Italian Southern Partnership for Advanced Computational Infrastructures), is currently used to support bioinformatics experiments on distributed and huge data banks. The framework provides a uniform access interface to access and interact with heterogeneous DBMS such us PostgreSQL, MySQL, Oracle, DB2, SQLite, etc.

8 Conclusions

GILDA represents a very valuable tool in the Italian INFN Grid Project [25] as well as in several Grid projects around the world. Indeed, its t-infrastructure has been used so far in more than 300 induction courses in 53 different countries all over the world. Altogether, more than 13000 certificates were issued by the GILDA Certification Authority, and at the time this article was written, more than a thousand users were currently registered with the GILDA Virtual Organization.

Through GILDA a broad range of users are able to get quick and easy access to a "real world" Grid environment. Furthermore, GILDA facilities are open to any organizations or educational institutions who want to adopt it in their Grid training programs.

In that sense, the main business benefits of adopting GILDA can be summarized as follows:

1. It is free;
2. No costs to leverage Grid resources are required to use it;
3. It is the first place where a user can takes experience on Grid Computing;
4. It provides an affordable and pleasant experience in Grid training;
5. It provides a good infrastructure where business applications can try "gridifying" operation exploiting the GILDA Team support.
6. It improves local research activity quality providing more power computation and storage resources;
7. It contributes to promote international cooperation and partnership.

Acknowledgments. The GILDA Team would like to thank all the partners (project and institutions) that collaborate on proving resources, contents, services and human effort to this project.

References

1. EGEE. (2008). Enabling Grids for E-sciencE. Retrieved February 8, 2009, from http://www.eu-egee.org/
2. EELA-2. (2008). E-science grid facility for Europe and Latin America. Retrieved February 8, 2009, from http://www.eu-eela.eu
3. EU-IndiaGrid. (2006). Retrieved February 8, 2009, from http://www.euindiagrid.org
4. EUMed-Grid. (2006). Retrieved February 8, 2009, from http://www.eumedgrid.org
5. EUChinaGRID (2006). Retrieved February 8, 2009, from http://www.euchinagrid.org
6. ICEAGE - International Collaboration to Extend and Advance Grid Education. Retrieved February 8, 2009, from http://www.iceage-eu.org
7. OMII-Europe. Retrieved February 8, 2009, from http://omii-europe.org/
8. BEinGRID - Business Experiments in Grid. Retrieved February 8, 2009, from http://www.beingrid.eu/

9. gLite – Lightweight Middleware for Grid Computing. Retrieved February 8, 2009, from http://glite.web.cern.ch/glite/
10. EGEE - Try the Grid. Retrieved February 8, 2009, from http://egee2.eu-egee.org/try-the-grid
11. GILDA Wiki. Retrieved February 8, 2009, from https://grid.ct.infn.it/twiki/bin/view/GILDA/WebHome
12. GILDA Support. Retrieved February 8, 2009, from http://gilda-support.ct.infn.it/
13. GILDA Virtual Services. Retrieved February 8, 2009, from https://gilda.ct.infn.it/VirtualServices.html
14. GILDA Grid Demonstrator. Retrieved February 8, 2009, from https://glite-demo.ct.infn.it/
15. GILDA Grid Tutors. Retrieved February 8, 2009, from https://gilda.ct.infn.it/grid-tutor.html
16. LCG Middleware. Retrieved February 8, 2009, from http://lcg.web.cern.ch/LCG/activities/middleware.html
17. Barbera, R., Falzone A. & Rodolico A.(2003, March). The GENIUS Grid Portal. Paper presented at Conference for Computing in High Energy and Nuclear Physics (CHEP 2003), La Jolla, California, USA.
18. Cirrone, G.A.P., Cuttone, G. et al. (2005) Implementation of a new Monte Carlo-GEANT4 Simulation tool for the development of a proton therapy beam line and verification of the related dose distributions. In: IEEE Transactions on Nuclear Science, Vol 52, Issue 1, Part 1, pp.262-265.
19. OpenGATE collaboration. Retrieved February 8, 2009, from http://opengatecollaboration.healthgrid.org
20. GILDA site installation. Retrieved February 8, 2009, from https://gilda.ct.infn.it/sites.html
21. Calanducci,A., Cherubino,C., Ciuffo,L.N., Scardaci,D. "A Digital Library Management System for the Grid", Fourth International Workshop on Emerging Technologies for Next generation GRID (ETNGRID 2007) at 16th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE-2007), GET/INT Paris, France, June 18-20, 2007 (http://etngrid.diit.unict.it/2007/index.html)
22. Unico Informatica S.R.L.. Retrieved February 8, 2009, from http://www.unicosrl.it/
23. INFN, Sezione di Catania. Retrieved February 8, 2009, from http://www.ct.infn.it/
24. TriGrid VL. Retrieved February 8, 2009, from http://www.trigrid.it/
25. INFN Grid. Retrieved February 8, 2009, from http://grid.infn.it/
26. GRelC - Grid Relational Catalog Project . Retrieved February 8, 2009, from http://grelc.unile.it/
27. Fiore S., Cafaro M. & Aloisio G. (2007, June) GRelC DAS: a Grid-DB Access Service for gLite Based Production Grids. Paper presented at the Fourth International Workshop on Emerging Technologies for Next-generation GRID (ETNGRID 2007), Paris, France

Grid Governance

Vladimir Dimitrov
Faculty of Mathematics and Informatics, University of Sofia,
5 James Bourchier Blvd., 1164 Sofia, Bulgaria
cht@fmi.uni-sofia.bg

Abstract. Success of Grid implementation in the enterprises strongly depends of Grid governance. The barriers in front of it and some recommendations how to overcome them, are subject of this paper.

Keywords: Grid Computing.

1 Introduction

The main Grid idea [1] is that computing is a utility. Grid user does not care where are stored his/her data or where is actually executed his/her computing requests. The user has to requests storage and/or computing and he receive them according his/her requirements.

From Enterprise Grid [2] point of view, Grid computing is IT infrastructure that meets business requirements for efficient resources utilization and for reduced expanses. The term Enterprise Grid Computing (EGC) is applicable in the boundaries of single enterprise. EGC assure that resources such as computing power, storage, database system, information and etc, are provisioned when and where they are requested. Such an infrastructure provisioning EGC is called Enterprise Grid. Enterprise Grid architecture has three main features: resources virtualization, dynamic resources provision, centralized resources management.

Virtualization separates resources from its consumers. Virtualization is a layer between resources and consumers. Virtualization allows underlying resource to be changed with similar one without affecting the consumer. Consolidation of all similar resources in a single global pool is the first step to virtualization. Term resource is applied here to physical resources as server, storage, database, and to abstract ones as information and application logic.

Typical enterprise architecture statically assigns resources to the applications taking in account their maximal needs. In an enterprise with virtualized resources, they can be provisioned when they are needed. When an application no more needs of a provisioned resource, it is returned back. Information can be dynamically shared. After the virtualization of information source, the consumer does not need to know which system store the information – it simply delivered at request.

Resource assignment in today enterprises is usually done manually with human intervention. In Enterprise Grid model resource provisioning is done automatically in accordance with politics preset by the administrators. Such an automatization is possible only when resources are virtualized and dynamic resource provisioning is available.

Important element of EGC is centralized IT resource management. It allows all Enterprise Grid resources to be controlled. Administrators use centralized management to control, monitor, add and remove resources as servers, storage, databases etc. Dynamic resources management is simplified when similar resources are managed from one place with common interface.

2 Transition to Grid

Enterprise Grid Computing can be implemented in three strategic steps: standardization, consolidation and automatization.

Transition to Grid will meet many barriers put by people, processes and technologies. For that reason pilot project has to be carefully prepared and implemented. This project has to demonstrate benefits of Grid and its Return of Investment.

3 Barriers in Front of Grid

Organizations feel themselves as owners of resources that they have bought or that have been assigned to them. They are afraid that if resources are not by them or not managed by them that they will not have access to these resources for their jobs. Organizations think that Grid with shared servers, storage and data will make data security weaker. Application users can be afraid that their applications will run at lower priority or with fewer resources than the other applications.

These political and organizational barriers are sometimes harder than the technological ones. The resistance to change could be very difficult to overcome.

Not every promise of Grid computing can be achieved right now and transition to grid to be done in one night. Some of main misunderstands about Grid computing are:

- It is possible to create Grid with mixture of different operating systems: Windows, Linux, Solaris etc. Nowadays applications easier share resources than the operating systems.
- Grid computing will utilize free desktop computers in the enterprise. Applications are written to run on specific server and it is impossible to move dynamically part of their calculations. In addition there are many security problems with some calculations that it is better not to move them.
- SOA is a panacea for integration of applications - wrap all applications in Web services and they will start to talk to each other and could create networks of dynamic business processes. Web services can easy represent applications as Web services, but all participating parts have to be synchronized by names, attributes and semantics to create networks of business processes.

Another problem is the lack of Grid standards, which means that software delivered by different vendors will not interoperate.

4 Implementation of Transition to Grid

Transition to Grid is a continuous process and requires long term engagement. That is why high management support is very important. This initiative needs of strong leader capable to attract for the idea different business departments, to help in solution of political and organizational problems and to resolve difficult situations that can appear during the transition to Grid computing.

One of the barriers in front of Grid computing is that business departments fear to lose control over their resources. To overcome this barrier they have to be convinced that via Grid they could receive more resources than those they have now. One such possibility is to be created Grid serving those applications that haven't enough resources and have to wait until suitable resources are released for them. This Grid can demonstrate how new applications are deployed without months of work. If information Grid would be created it is needed to discuss with potential users the value of consolidated information.

Human resistance can be overcome with some of the following approaches:

- **Effective communication**. It is important to be created direct and open communication with the management to clarify to the people the positive impact of the change to the organization. The most important is people to understand their place in the new organization and business organization to understand that their needs will be satisfied in the new model.
- **Team building**. The team has to be build of representatives from all business departments to motivate them to work for the common idea. Grid is intended to serve all business departments and administrative resources of all these departments have to be involved in the efforts for Grid building.
- **Time for adoption**. A time is needed before the people start to use the new concept and a time is needed before business departments start to use efficiently Grid computing. Only when the organization achieves enough experience with Grid model, only then the organization can start optimizing the Grid. For example, in Grid model there is no need for business departments to install software updates and patches. Instead, Grid administration has to install software updates and patches, after a careful testing proving that they satisfy enterprise standards. This is a new social behavior impacted by Grid computing. The resistance on this approach can be argued that much time is spent for this procedure and new constraints are applied to the business. Business departments have to understand that this change to standardization simplifies configuration management of the systems and the result would be improved fighting with software problems.

It is better to start with something small, to start with something that could be widely accepted in the whole organization. It can be characterized with:

- minimal resistance;
- benefits have to be measured even in small scale;
- results can be achieved in short times.

Examples:

- If the enterprise intends to buy new hardware resources, then cheap module servers could be bought and organized in a cluster. Later, databases and applications could be consolidated on it.

- If the organization has problems with its resources management, then centralized system for resources management could be bought. In such a way currently available administrators will manage enlarging IT infrastructure.
- If the enterprise has data intensive applications (such for business analyses) that need of more database resources, then processing could be moved to database system with more server resources.
- If the enterprise wants to achieve modularity in software development or wants to improve business processes dataflows, then Service-Oriented Architecture and Web services could be used.

After the Grid benefits have been demonstrated, new resources and applications could be moved to it increasingly scaling the Grid.

5 Strategic Steps to Enterprise Grid Standardization

Standardization of technology means shortening the number of heterogeneous products and the number of software and hardware vendors. Standardization is applied to architecture of the products and to the all IT life cycle processes: development, delivery, support and decommission. When vendors are selected preferable ones are those ones whose products supports currently available industry standards.

The enterprise has to support small number of standard products. Every product has to be tested in centralized IT department before its acceptance for use in the enterprise. In such a way testing expenses in the different IT departments in the enterprise are shortened. The centralized testing ensures that the product passes through more precise testing.

Standardization is the fundament of interoperating and consolidated applications. Enterprises have to use available standards in application development. An application that follows the standards has longer life. It is expected that applications developed with today industry standards would live in the future Grid environment. Such technology standards are J2EE and Web services.

Standardization ensures that on selection phase only products suitable to the enterprise would be selected.

During the development and quality assurance, the standardization ensure that repeatable development processes are followed and needed quality of the product is achieved via enough testing before the mass product acceptance. During the deployment, standardization ensures that the environment is the same as that one used for quality testing. In production phase, standardization ensures that repeatable and consistent procedures are followed for the regular support of the production environment. During decommission, standardization ensures that the components are secure decommissioned, for example secure information from commissioned component cannot be stolen.

6 Consolidation

After the standardization is the consolidation. Instead of many distributed IT environments, enterprises will support a few data centers. Consolidated data center benefits from the centralized management of big number of components. Not only expenses for electricity, cooling, space requirements are shortened, but also overall expenses for IT infrastructure management.

Enterprises no more need of separate server or database system for every application. Several applications can use one database system and one application server. Consolidation means shortening the number of database systems used at the enterprise.

Enterprises have to consolidate its hardware infrastructure. Blade servers are very appropriate for this. SAN and NAS are suitable for storage consolidation. These technologies unify storage infrastructure and support the integration providing suitable level of QoS for different applications.

Data integration could be supported at physical level if the data are collected in one database or at logical level with data hubs. Common interface to data for all applications virtualizes information source from the applications and allows future migration to new application architecture like SOA.

7 Automatization

After the standardization and consolidation is the automatization. The former one decreases overall expenses for IT management. Automatization decreases the risk of human errors. Enterprise architects have to look for all possibilities for IT automatization in their strategic planning.

Enterprises can find automatization elements in every IT cycle like:

- Automatization of initial server inclusion;
- Automatization of the maintenance.

Regular maintenance tasks like backups could be subject of automatization. If IT systems use a few configurations, then the same scripts could be used in these environments. Exceptions or incidents will be reported the administrators to take corrective actions.

Acknowledgments. This work is funded under the contract 200/15.05.2009 “Development of Grid Infrastructure for Research and Education” with Scientific Research Fund of University of Sofia.

References

1. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
2. Enterprise Grid Alliance, Open Grid Forum, http://www.ogf.org/

Adapting ESB Infrastructure for Grid

Radoslava Goranova

Faculty of Mathematics and Informatics,
1164, 5 James Baucher, Sofia, Bulgaria
{radoslava@fmi.uni-sofia.bg}

Abstract. Enterprise service bus (ESB) is a software framework for integration that enables service-oriented architecture for non service-oriented systems. Grid from the other hand is an upcoming technology for inexpensive and consistent access to computational and storage resources. The progress of the Grid as technology is up to the level, where development of Grid applications, toolkits and portals is tightly dependable from underling Grid environment. Applying service-oriented approach for integration in grid environment can provide standardized platform for integration and for development of reusable services. In this article we describe our experience with adaptation of two integration environments: ServiceMix and WSO2 ESB in g-Lite Grid middleware.

Keywords: Grid, ESB, ServiceMix, WSO2 ESB, SOA, SOI, g-Lite.

1 Introduction

Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high-end computational capabilities [1]. Advantages of this technology are the direct access to geographically distributed resources, "unlimited" use of CPU power, standard protocols for transfer and data calculation and common use of shared data and software.

The Grid is infrastructure for coordinate use of distributed resources, shared by different institutes, computational centers and organizations. This new way of using computers, networks and devices has great benefits, for every scientist who needs unlimited computational power, unlimited storage resource or fast solving of difficult mathematical, chemical or physical problem.

From architectural point of view Grid is five-layered (Figure 1). Grid architecture is constructed by fabric layer, connectivity layer, collective layer, application layer and user application layer. Fabric layer provides resources - computational, storage systems, network resources, specific devices, sensors even clusters or pools. Connectivity layer defines communication and authentication protocols for exchanging data between fabric layer and secure mechanism for resource access. Collective layer contains protocols and services for resource interaction monitoring, diagnostics, scheduling, brokering, service discovering and others. Application layer provides software tools: application toolkits, portals and libraries for access to these services defined and implemented in collective layer. User application layer covers user defined applications and user defined grid services.

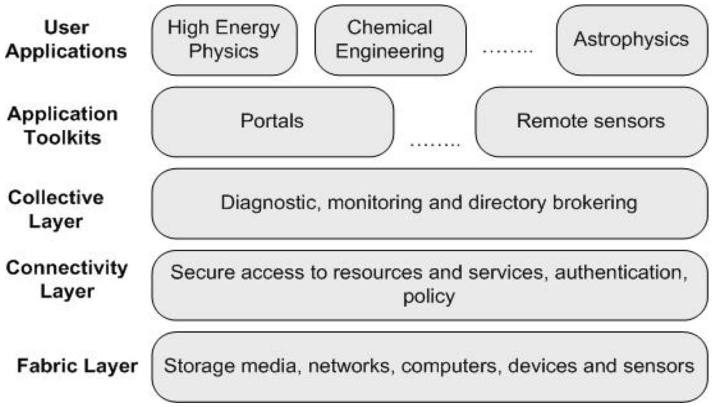


Figure 1. Grid architecture

According to the way how Grid services are exposed we can classify three type of Grids: service-oriented, partially service-oriented and non service-oriented. Service-oriented girds, provide grid services as services in the way they are defined in SOA.

In service-oriented architecture (SOA) [2], services possess the following characteristics:

- Services can be individually used or integrated in composition of services.
- Services communicate by message transfer.
- Services can be part from workflows.
- Services can be self-contained or they may depend on the availability of outer resources or other services.
- Service provides information for their interfaces, policies and communication they support.

To avoid the confusion between terms services and grid services, we defined what we understand behind these terms.

By the term service in the way they are defined in SOA we mean software component that realizes specific system’s logic and provides it to the end client by interfaces. The interfaces describe a set of public operations for each service. Interfaces are self-describing and platform independent. The definition of these software components follows the principle of service-orientation [3]

By grid services we mean the grid resources and components which provide main grid functionalities as storage, information for grid resources and computational capabilities. Grid services can also be services, but not necessarily, it depends of the way the services are designed and how they are exposed.

Until recently all innovations around the Grid was in building grid middleware, which to provide, all features and advantages mentioned above. To build a Grid middleware is a great challenge, because of all security issues related with accessing resources from different institution or organization. However, nowadays the accent of the Grid innovations is to optimize grid middleware in such a way that it provides not only clients and tools for resources' access, but also high-level services. That suppose, grid functionalities to be exposed as services, undependable units which capsulate the smallest functionality which is possible. Building grid services in the sense of service-oriented architecture requires following the principles of service-orientation. For existing Grid environments, however this is not an easy task because of the existing legacy functionality. Here comes the place of service-oriented integration approach.

2 ESB and the Grid

The most common problems with access to grid middleware are related with non-trivial and non-unified way for access to grid services. The IT practice shows that for solving similar type of problems, ESB approach for integration can be applied.

The ESB approach provides pluggable services and standard infrastructure for integration. As an integration infrastructure, the ESB provides a number of functions including routing, transformation, content-based routing. An ESB also supports requirements as security and orchestration.

The ESB can be defined as integration platform that provides execution environment for management of messages, web services and data transformation. ESB also is service-oriented methodology for application integration. One of advantages of ESB is that it enables service-orientation for already developed systems. Attributes of an ESB integration infrastructure are message-based - to promote loose coupling and open standards-based.

If we reconsider the Grid architecture described above, the place of ESB (Figure 2) is between collective layer and application layer. The reason is that most of the grid middleware implements first three layer form the Grid architecture. Once of them also provides well defined services, which can be easily used from upper layers, but most of them do not provides such services. And here is the role of ESB. ESB in Grid [4] has to provide integration layer of services, which to follow the principles of SOA. Also ESB in Grid has to manage access of application to legacy grid services and to standardize the access to these services.

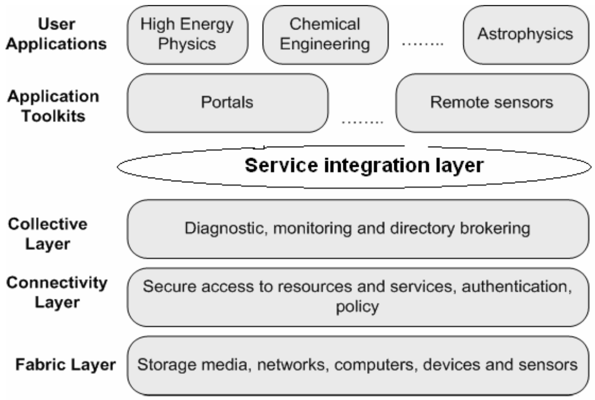

User Applications
High Energy Physics
Chemical Engineering
....
Astrophysics
Application Toolkits
Portals
....
Remote sensors
Service integration layer
Collective Layer
Diagnostic, monitoring and directory brokering
Connectivity Layer
Secure access to resources and services, authentication, policy
Fabric Layer
Storage media, networks, computers, devices and sensors

Figure 2. ESB and Grid architecture

There is no common standard for Enterprise Service Bus. The reason is that every provider has its own definitions, based on the technologies which use and the products which develop. However, the most spread are realizations of Oracle, IBM, Mule, JBoss, ServiceMix and Apache Synapse. For the aims of our investigation we try to adapt implementation of JBI - ServiceMix and Apache Synapse - WSO2 ESB in g-Lite grid environment.

g-Lite [5] is a Gird middleware, which provides grid services as data services, security services, job management services and information services (Figure 3).

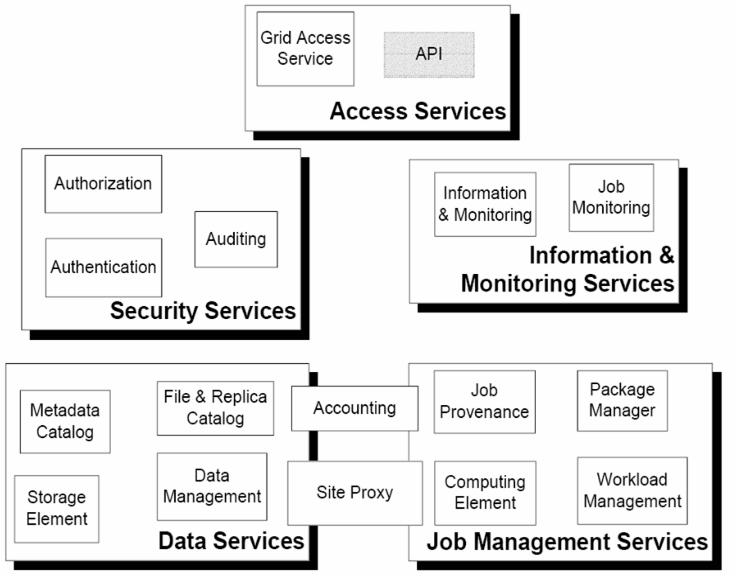


Figure 3. g-Lite services [5]

The middleware is consisting of different components, which provide functionality for storage, computation, resource discovery and security mechanism. Once of these components expose their functionality as services, like WMS or FTS, the other like information system BDII does not. Disadvantage of provided services is that it is not used common technology for exposition of these services. Once of them are implemented by Axis1.4, others by Axis2, and third just provides CLI on C, Java or Pyton.

2.1 JBI and ServiceMix ESB

Java Business Integration (JBI) [6] is a Java-based standard for service interaction and mediation of message exchange between the different components. JBI (Figure 4) uses the principle of message normalization in such a way, so every message incoming to the environment is transformed to common format used in the environment. The component which does this transformation is called NMR

(Normalized Message Router). This component is a core part of the integration environment and its role is to proxy messages between different plug-ins. The main JBI components are binding components and service engines.

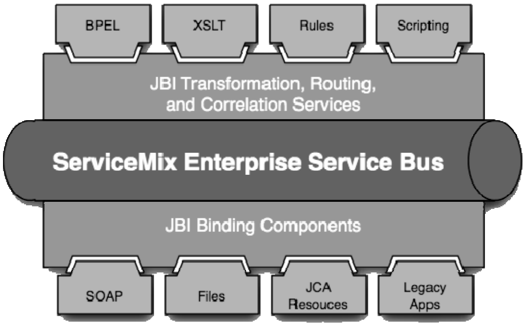

BPEL
XSLT
Rules
Scripting
JBI Transformation, Routing, and Correlation Services
ServiceMix Enterprise Service Bus
JBI Binding Components
SOAP
Files
JCA Resouces
Legacy Apps

Figure 4. ServiceMix [7]

Binding components are used for communication with external services, for normalization and de-normalization of messages and for communication with the different protocols such as HTTP/S, JMS, FTP, SMTP.

Service Engines implements specific logic in the environment, as BPEL engines and communicates only with the NMR. Communication with the external services is by binding components.

One implementation of JBI is ServiceMix. ServiceMix also provides an already developed component which covers the most common used functionality in the different systems.

2.2 Apache Synapse and WSO2 ESB

Apache Synapse [8] is an ESB that has been designed to be simple to configure, very fast, and effective at solving integration problems. Synapse has support for HTTP, SOAP, SMTP, JMS, FTP and file system transports. Apache Synapse (Figure 5) has the following basic components proxy services, registry and message mediation.

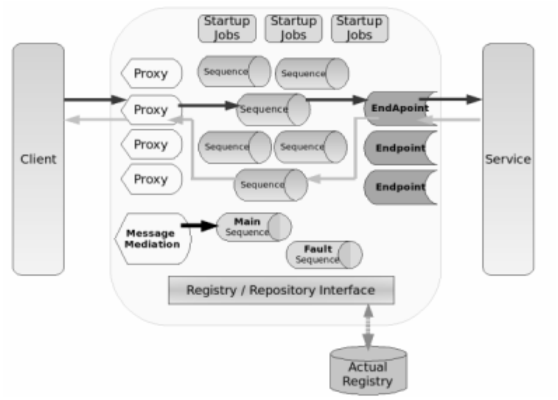


Figure 5. WSO2 ESB (Apache Synapse) [9]

Proxy services are services build-in Apache Synapse, which are accessible from outside, but provides access to services which are not accessible form outside, like legacy services. The proxy service acts as a service hosted in Synapse, and typically fronts an existing service endpoint. A proxy service can be created and exposed on a different transport or schema.

Synapse has a completely asynchronous core, and supports non-blocking HTTP and HTTPS using the excellent Apache HttpCore module. Built in Registry/Repository, facilitating dynamic updating and reloading of the configuration and associated resources. It stores configuration files and resources. Message mediators save, transform and route a given message on the base of it content.

2.3 ServiceMix, WSO2 and g-Lite

For the experiment, the two environments were installed on g-Lite middleware. Installation and configuration process for two of them was trivial and follows software documentation. The both environments ServiceMix and WSO2 present communication and message exchange on very low level which is an advantage if we develop our system from the scratch but for already developed infrastructure it is not.

An example WSO2 ESB Mediator was developed, but some unresolved issues related with client certificate occur. It seems like WSO2 ESB does not provide mechanism to extract information from client's X509 certificate during mediator invocation. As we already mentioned in Grid it is essential feature to be able to extract user DN. On the base of users' certificate can be implemented different behavior of the service. Another cons for WSO2 is provided registries. On first look it makes impression that registries stores service description, but their functionality is just to store external files needed from developed services.

With ServiceMix the things are almost the same. The environment provides mechanism for connection between different g-Lite components on low level, which concern message transfer. On one way or another g-Lite already provides such communication between the different components.

We may conclude that the process of installation and configuration of the two ESB environments are less challenge, then their adaptation to already existing Grid middleware. The difficultness comes from Grid specification in which security plays vital role. Undoubtedly, the both ESBs have advantages as flexibility and simplicity, but their adaptation to g-Lite will require much more efforts then development of new integration solution.

Future work

To solve the problem for unified access to grid services, integration layer of services has to be developed. This layer has to provide services for unified access to grid services, mechanism for users' services definition and deployment, registry with defined services and functions for access to this registry.

Acknowledgements

This paper is supported by the Sofia University SRF under Contract N 7/2009.

References

1. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
2. Erl, T.: Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall PTR (2005)
3. Service-oriented architecture principles [http://www.soaprinciples.com/];
4. R. Goranova, ESB in Grid, International Conference on Computer Science'08, Kavala, Greece, 2008.
5. Programming the grid with g-Lite, http://personal.cscs.ch/~pkunszt/programming.pdf
6. JBI, http://servicemix.apache.org/what-is-jbi.html
7. ServiceMix ESB, http://servicemix.apache.org/home.html
8. Apache Synapse ESB, http://synapse.apache.org/
9. WSO2 ESB, http://wso2.org/project/esb/java/1.7/docs/ESB_Configuration_Language.html

Comparison of Grid Resource Discovery Approaches

Georgi Pashov [1,5], Kalinka Kaloyanova [1]

[1] Faculty of Mathematics and Informatics, Sofia University “St. Climent Ochridsky”, 5, James Bourchier Blvd., 1164 Sofia, Bulgaria
{gpashov, kkaloyanova}@fmi.uni-sofia.bg

Abstract. Matching the requirements of an application with available resources is one of the key aspects of Grid systems. The challenge is to devise highly scalable and fault tolerant Grid resource discovery techniques. In this paper we propose taxonomy of Grid resource discovery requirements in order to provide objective and uniform criteria for evaluation and comparison of resource discovery approaches. We analyse some of typical Grid resource discovery approaches and finally present a qualitative comparison of these approaches based on their functionality, scalability, reliability, performance, etc.

Keywords: Grid resource discovery.

1 Introduction

An efficient resource discovery mechanism is a basic aspect for successful implementation and broad adoption of a Grid infrastructure, as it provides transparent global resource view for job scheduling in Grid. Resource discovery is a systematic process of determining which Grid resource is the best candidate to complete a job with the following trade-offs:

- In shortest amount of time;
- With most efficient use of resources;
- At minimal cost [1].

Compared with the resource discovery in traditional distributed and cluster systems, detection of suitable resource in Grid environment is more complex as resources are globally distributed, heterogeneous in nature and owned by different individuals and organizations, each having their own management and access policies and cost model [2]. Moreover, Grid resource discovery should be scalable, fully decentralized from any global control and should tolerate intermittent resource participation.

The motivation behind this paper is to study the mechanisms for resource discovery in Grid environment and to determine which ones are the most appropriate for fast, efficient, reliable and cost effective resource location in global-scale Grid. Key aspect for accomplishing this task is to categorize Grid resource discovery

[5] This paper is supported by the Sofia University SRF under Contract N 7/2009.

requirements, which will provide objective and uniform criteria for evaluation and comparison of Grid resource discovery approaches.

2 Taxonomy of Grid Resource Discovery Requirements

In general, Grid resource discovery requirements can be classified as *functional* and *non-functional* (Fig. 1). Functional requirements are associated with services, tasks or functions the system is required to perform, while non-functional ones define the overall qualities of the system such as reliability, scalability, performance, etc. Functional requirements drive the application design of a system, while non-functional requirements drive the technical architecture of a system.

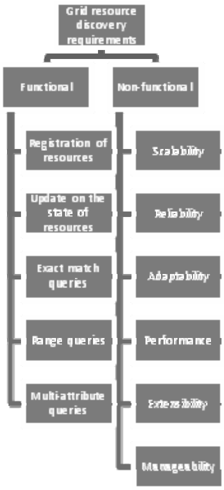


Fig. 10. Taxonomy of Grid resource discovery requirements

Some of the main functional requirements that Grid resource discovery systems should support are: *registration of resources* with their characteristics, *update on information about the state of resources*, *exact match queries* (for example, machines running specific operating system), *range queries* (machines with at least 256 MByte of memory, for instance), and *multi-attribute queries* (include conditions upon several attributes).

Non-functional requirements, like scalability, reliability, adaptability, performance, etc., are essential because first and foremost Grid is a technical challenge. *Scalability* is the ability of the system to maintain usability in distributed geographic area with exponentially growing number of participants and resource discovery requests. *Reliability* is defined as resistance to system failures. *Adaptability* refers to the degree to which adjustments in overlay search structures and request execution are possible in response to or in anticipation of changes in environmental conditions. *Performance* is characterized by the ability to execute large number of requests in reasonable time at minimal cost and to handle frequent updates on the state of resources. *Extensibility* is a measure of the ability to extend a system with new functionality and the efforts required to implement this extension. *Manageability* defines the cost for system maintenance.

3 Grid Resource Discovery Approaches

3.1 Globus Toolkit's Monitoring and Discovery Service (MDS4)

The Globus Toolkit's Monitoring and Discovery System (MDS4) is a suite of components and services for monitoring and discovering resources on Grid. MDS4 architecture is based on "protocol hourglass", concept in which diverse schemas, supplied by different local information providers (figuratively bottom of the hourglass) are translated into a unified XML schema (using GLUE Schema for Grid resource modelling whenever possible) and as a result various tools and applications (top of the hourglass) can use uniform Web service interfaces for resource's enquiry, subscription and notification.

MDS4 provides two higher-level services: *Index service*, which collects and publishes aggregated information about Grid resources, and *Trigger service*, which collects information and performs actions when certain conditions have occurred. The Index and Trigger service implementations are both built on the *Aggregation Framework*, a common software framework for collecting, aggregating and maintaining data. Services built upon Aggregation Framework collect information through WSRF-compliant (Web Service Resource Framework) services or through domain-specific executables.

MDS4 Index Services have a number of features related to Grid scalability issues [3]:

- Each virtual organization (VO) may maintain one or more Index services to register resources;
- Index Services can be arranged in hierarchies enabling the consolidation of data from multiple locations, but there is no single global Index that provides information about all resources on the Grid.

3.2 Peer-to-Peer-based Approach

In recent years Peer-to-Peer (P2P) networks have emerged as one of the most successful ways of sharing files and computational power in distributed and fault tolerant way. P2P paradigm is particularly appropriate for Grid resource discovery because of its global scale, peer autonomy and fault tolerance.

In [4] Iamnitchi et al. propose a fully decentralized P2P architecture for resource discovery in Grid environment. The authors partition a general resource discovery solution into four components: *membership protocol*, *overlay construction*, *preprocessing*, and *request processing*. The membership protocol specifies how new nodes join the network and how nodes learn about each others. The overlay construction function selects the set of collaborators from the local membership list. Preprocessing refers to techniques used to enhance search performance prior to executing requests (as dissemination of resource description for instance). The request processing defines how requests for resource information could be executed. The request processing function has two components: local, that looks up a request in local information, and remote, that implements the request propagation rules.

Unstructured P2P. Unstructured P2P systems are characterized by the lack of an underlying organizational structure. Each node in the network is connected to a subset of neighbour nodes, without any rule or condition that can lead to a general overlay structure of the network. The unstructured nature of these networks implies that there is no deterministic information about resource location in the system. Therefore the prevailing request propagation method is controlled flooding – a process in which requests are forwarded in overlay until their Time-to-Live parameter is exhausted.

In [4] Iamnitchi et al. examine four different strategies for request propagation: *random walk* (the nodes to which a request is forwarded are chosen randomly), *learning-based* (request is forwarded to nodes that have already executed a similar requests), *best-neighbour* (for every node the number of received answers are recorded and request is forwarded to these that answered a largest number of queries), *learning-based* + *best-neighbour* (similar to learning-based strategy except that when no statistics are available the best-neighbour approach is applied).

In [5] Mastroianni et al. adopt the *super-peer* model to design a P2P-based Grid information service in which only part of the peers, usually these with the highest bandwidth, are connected in an overlay network. These nodes, called super-peers, act as a centralized resource index server for an organization. The rest of the nodes are connected to a super-peer. Super-peer model is broadly exploited in Grid because it is naturally suitable for large-scale Grid environments. The resource discovery protocol works as follows. A query is sent to the local super-peer, which is looking for requested resources locally. If resources are found response is sent back. Otherwise the query is forwarded to a subset of neighbours, determined upon statistics from previous executions, which in turn request underlying systems and so on.

Structured P2P. Structured P2P systems use a rigid structure to organize an efficient search overlay network. The basis of this structure is distributed indices usually based on *Distributed Hash Tables* (*DHT*) or binary trees. The structured P2P systems are applicable only for ad hoc resource requests. The disadvantages are that maintenance of the overlay structure is too expensive and that they do not support more general queries.

Chord is the first structured P2P system [6]. Both peers and values of resource characteristics are mapped through the same hash function to *m*-bit key space. Peers are ordered in circle according to their keys and each of them is connected to other peers on logarithmic increased distance. Each peer keeps index of resources whose keys belong to the range between the key of the predecessor and its own key.

In [7] MAAN, an extension on the Chord protocol supporting multi-attribute range queries, is proposed. In MAAN a separate Chord overlay network, using locality preserving hash function, is created for each attribute. Multi-attribute queries can be executed in two ways. The first one is iterative: query is divided into *M* sub-queries, each on single attribute; next all M sub-queries are executed and results are intersected. The second method is more efficient, although more complex: at first the sub-query for the least selective attribute (attribute with minimum expected matching results) is executed; then found resources are checked against the rest sub-queries.

3.3 Semantic-based Approach

In [9] Tangmunarunkit et al. propose a flexible and extensible approach for performing Grid resource selection using an ontology-based matchmaker. In traditional resource matchmaking, common attribute names are used for both, resource characteristics description and request requirements specification. During the matchmaking process the resource's attribute values, advertised by resource providers, are compared with the respective attribute values required by resource consumers. For the comparison to be meaningful and effective, the resource providers and consumers have to agree upon attribute names and values. In a heterogeneous multi-institutional environment such as the Grid, it is difficult to enforce the syntax and semantics of resource descriptions and this makes such systems inflexible and difficult to extend with new characteristics and concepts. Semantic-based approach tries to overcome these disadvantages.

In semantic-based approach separate *ontologies* are created to declaratively describe resources and requests using an expressive ontology language. The loose coupling between resource and request descriptions removes the tight coordination between resource providers and consumers. In addition there are two more components in ontology-based matchmaker: *domain background knowledge*, that captures additional knowledge which cannot be expressed by the ontology language (for example which operating systems are compatible with each other); and *matching rules* between request specification and resource capabilities. In order the matchmaking process to be effective it is necessary the resource matchmaker to allow flexible matching; and both providers and consumers to be precise with description of their resources and requests.

3.4 Parameter-based Approach

A new concept for *Grid potential* is proposed in [10]. Informally, Grid potential at a node in the Grid can be considered as the computational power that can be delivered to an application at that node. This potential drops off, depending on network characteristics, as we move away from the node along the Grid. The authors examine various strategies for maintenance the consistency of distributed resource status databases through data dissemination: *universal awareness*, *neighbourhood awareness* and *distinctive awareness*. In universal awareness status information is disseminated unlimited across the whole network. For large scale networks this approach causes significant communication overhead. In neighbourhood awareness status information reached only nodes that are at most on fixed distance away from originating node. In this approach dissemination overhead is limited but nodes that are behind the dissemination horizon could not have information for remote nodes. The main idea behind distinctive awareness is as follows: information on rare resources is disseminated over a longer distance and thus a greater set of nodes are aware of such resources; while information on frequent resources is spread over limited space, as such resources, being widespread, can be found even in short distance. If all nodes are homogeneous, distinctive awareness algorithm is like neighbourhood awareness algorithm. In a highly heterogeneous Grid, this algorithm should deliver resource discovery efficiency close to a universal awareness while having a communication complexity closer to the neighbourhood awareness algorithm.

3.5 QoS-based Approach

Quality of Service (*QoS*) in Grid is mechanism for increasing the efficiency of user collaboration in using Grid resources based on quality criteria such as priority, fairness, and economic gain [11]. The main requirements to guarantee QoS in Grid are: supporting mechanisms for advance or 'on-demand' resource reservation, supporting reservation policies governing when, how, and who can use resources, supporting Service Level Agreements (SLAs) that assure the clients of the resource quality they expect during the service session.

In [12] is addressed the problem of resource discovery in a Grid based multimedia environment, where the resources providers are intermittently available. The authors present a generalized algorithm for discovering intermittently available resources (DIAR) by modelling the DIAR problem using a graph-theoretic approach. Various policies for QoS-based resource discovery that can meet a variety of user needs are formulated. The performance of QoS policies based on different time-map scenarios and placement strategies in a centralized system is evaluated. Performance results illustrate the added benefits obtained by adding flexibility to the scheduling process.

3.6 UDDI-based Approach

In [13] Benson et al. evaluate using of UDDI, the Universal Description, Discovery and Integration framework for resource discovery in Grid. UDDI provides a means of advertising and browsing web service information through four data structure types: businessEntity, businessService, tModel, and bindingTemplate.

The authors propose workarounds for some limitations of UDDI which are an obstacle to resource discovery in Grid: a lack of explicit data typing for information in the UDDI directory, difficulties in handling dynamic information that requires frequent updating and the limitations of the UDDI query model. The authors evaluate performance of a UDDI as a provider of resource discovery services using three metrics: system load on the UDDI hosting machine, mean update time for service provider information, and mean query time experienced by Grid users. The final conclusion is that "UDDI is only appropriate for small Grids in which scalability and precise performance reporting is secondary to the industry support and ease of installation that accompany this technology" [13].

In [14] has been proposed a DHT based UDDI Registry framework to support large scale discovery. The architecture enables organizations to keep autonomous control over their UDDI registries and at the same time allowing clients to query multiple registries simultaneously. The authors propose to build a DHT based overlay network of UDDI registries, where the DHT acts as a rendezvous network that connects multiple local registries. Each local UDDI registry has a proxy that mediates between it and the DHT Service. Initial prototype testing shows that the proposed architecture can support effective distribution of UDDI registries thereby making UDDI more robust and also addressing its scaling issues.

4 Comparison of Grid Resource Discovery Approaches

Comparison between the Grid resource discovery approaches based upon various functional and non-functional criteria are presented in Table 1 and Table 2 respectively.

Table 7. Comparison of Grid resource discovery approaches based on functional requirements.

Criteria	MDS4	Structured P2P	Unstructured P2P	Semantic	Parameter	QoS	UDDI
Registration and update on resources	Information is recent but not absolute latest	Information is recent but not absolute latest	Information is latest	Information is latest	Information is recent but not absolute latest	Information is latest	Information is latest
Exact match queries	Resources will be found if they exist	Resources will be found if they exist	Resources may not be found, even they exist, due to the strategies for limitation of flooding	Results depend on matching criteria (false positive/false negative)	Resources may not be found, even they exist, in neighbourho od awareness	Resources will be found if they exist	Resources will be found if they exist
Range queries	Not supported	Supported – using locality preserving hashing in DHT or tree indices	Not supported	Not supported	Not supported	Not supported	Workaround – mask values and apply wildcard-based searching
Multi-attribute queries	Not suitable – separate search for each attr. in global scale environment	Exists approaches designed for efficient multi-attr. search	Not suitable – separate search for each attr. in global scale environment	Supported	Not suitable – separate search for each attr. in global scale environment	Supported	Supported

The comparison indicates that super-peer, or *Cluster-To-Cluster*, approach covers in greatest extent requirements for Grid resource discovery since it combines efficiency of searching in centralized registry and load balancing, fault-tolerance and domain autonomy provided by distributed systems. The super-peers are interconnected in unstructured P2P network which is very suitable for handling frequent updates of resource metadata and allows execution of more general requests. A number of informed search strategies, such as Grid potential, routing indices, etc., can be used to increase the effectiveness of discovery process. If the majority of requests concern the values for a few attributes it is worth to consider whether to create distributed P2P indices on the most commonly used attributes. This approach has some disadvantages as well, one of which is that it is inflexible and difficult to extend with new characteristics and concepts. Symmetric description of resources and requests requires coordination between resource providers and consumers which is not easy task in global Grid.

Table 2. Comparison of Grid resource discovery approaches based on non-functional requirements.

Criteria	MDS4	Structured P2P	Unstructured P2P	Semantic	Parameter	QoS	UDDI
Scalability	Scalable – Index Services (IS) can be arranged in hierarchies	Good geographic scalability and limited traffic load	Good geographic scalability but big traffic load (flooding)	Limited scalability because of centralized matchmakin g broker	Scalable due to Grid potential concept used for metadata discovery	Uses different time map strategies to increase the scalability	Limited scalability because of centralized UDDI registry
Reliability	Fault tolerant – each VO maintains its own IS; Failure of one IS does not affect the other	Fault tolerant as it is a distributed system	Fault tolerant as it is a distributed system	Poor reliability – centralized ontology database is a single point of failure	Fault tolerant as resource information is distributed across all nodes	Poor reliability – centralized matchmakin g system	Poor reliability – centralized UDDI registry is a single point of failure
Performance on registering and updating	Very good – information for updates is not propagated to other nodes	Good – changes in resource characteristic s' values enforce index reorganizatio n	Very good – flooding – no need of metadata distribution; routing indices – need refreshing	Very good – updates are executed in centralized registry	Very good – disseminate information across network and recalculate nodes' potential	Very good – updates are executed in centralized registry	Very poor performance with frequent updates
Performance on querying	Good – the ISs allow last value of each data element to be cached in order to improve search performance	Good performance with limited communicati on overhead	Not very good performance with a huge communicati on overhead; Limitation of flooding: TTL, etc	Very good performance provided by centralized system	Good – distinctive awareness – resources are found while number of exchanged messages is limited	Very good performance provided by centralized system	Very good performance provided by centralized system
Extensibility	Good – coordination	Difficult to add new	Good; Coordination	Easily add new	Difficult to add new	Easily add new	Easily add new

	about exact meaning of new characteristics is required	characteristics; Coord. is required; Difficult impl. in global scale environment	about exact meaning of new characteristics is required	characteristics or concepts – no coordination is required	characteristics; Coordination about exact meaning is required	characteristics in centralized registry; Coordination is required	characteristics in centralized registry; Coordination is required
Manageability	Easily within a GT4 container	Difficult – complex architecture	Difficult – complex architecture	Quite easy – centralized ontology-based matchmaker	Manage consistency by using data dissemination algorithms	Quite easy – centralized QoS registry	Quite easy – centralized UDDI registry

5 Conclusions

The Grid resource discovery problem is a problem for finding the most appropriate resources for job execution. The task is complicated by the fact that the resource discovery is taking place in a global-scale environment with hardly predictable behaviour of the participants and frequent changes on the state of resources. This requires Grid resource discovery techniques to provide highly scalable and fault-tolerant mechanisms for efficient utilization of resources at minimal cost. We propose taxonomy of Grid resource discovery requirements in order to provide objective and uniform criteria for evaluation and comparison of Grid resource discovery approaches. We analyse some of typical Grid resource discovery approaches on the basis of their functionality, and quality factors such as scalability, reliability, performance, etc. Our assumption is that super-peer, or Cluster-To-Cluster, approach is the most suitable for Grid resource discovery system since it combines efficiency of centralized searching with scalability and reliability of distributed systems. Our future work aims at implementing simulators for different Grid resource discovery mechanisms upon super-peers, and comparing their performance results to conclude in the most appropriate solution for Grid resource discovery.

References

1. Sharma, A., Bawa, S.: Comparative Analysis of Resource Discovery Approaches in Grid Computing. Journal of Computers, vol. 3, no. 5 (2008)
2. Buyya, R., Chapin, S., DiNucci, D.: Architectural Models for Resource Management in the Grid. Springer Verlag LNCS Series, Germany, Bangalore, India (2000)
3. Schopf, J., Raicu. I., Pearlman, L., Miller, N., Kesselman, C., Foster, I., D'Arcy, M.:Monitoring and discovery in a Web services framework: functionality and performance of Globus Toolkit MDS4, Technical Report, Mathematics and Computer Science Division, Argonne National Laboratory (2006)
4. Iamnitchi, A., Foster, I., Nurmi, D.C.: A peer-to-peer approach to resource location in grid environments. Proceedings of 11th IEEE International Symposium on High Performance Distributed Computing (2002)

5. Mastroianni, C., Talia, D., Verta, O.: A Super-Peer Model for Building Resource Discovery Services in Grids: Design and Simulation Analysis, Proc. European Grid Conference (EGC 2005), LNCS, vol. 3470, 132-143, Springer (2005)
6. Stoica, I., Morris, R., Karger, D., Kaashoek, M., Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications, Proc. ACM SIGCOMM 2001 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication, 149-160 (2001)
7. Cai, M., Frank, M., Chen, J., Szekely, P.: MAAN: A Multi-Attribute Addressable Network for Grid Information Services, Proc. 4th Int. Workshop on Grid Computing (GRID 2003), 184-191 (2003)
8. Trunfio, P., Talia, D., Fragopoulou, P., Papadakis, C., Mordacchini, M., Pennanen, M., Popov, K., Vlassov, V.: Peer-to-Peer Models for Resource Discovery on Grids, CoreGRID Technical Report, TR-0028 (2006)
9. Tangmunarunkit, H., Decker, S., Kesselman, C.: Ontology-based Resource Matching in the Grid–The Grid meets the Semantic Web, Proceedings of the Second International Semantic Web Conference, Sanibel-Captiva Islands (2003)
10. Maheswaran, M., Krauter, K.: A Parameter-based Approach to resource discovery in Grid computing systems, 1st IEEE/ACM International Workshop on Grid Computing (2000)
11. Al-Ali, R., von Laszewski, G., Amin, K., Hategan, M., Rana, O., Walker, D., Zaluzec, N.:QoS support for high-performance scientific Grid applications, IEEE International Symposium on Cluster Computing and the Grid, 134- 143,(2004)
12. Yun Huang; Venkatasubramanian, N.: QoS-based resource discovery in intermittently available environments: Proceedings. 11th IEEE International Symposium on High Performance Distributed Computing, 50-59, (2002)
13. Benson, E., Wasson, G., Humphrey, M.: Evaluation of UDDI as a provider of resource discovery services for OGSA-based grids. 20th International Parallel and Distributed Processing Symposium (2006)
14. Banerjee, S., Basu, S., Garg, S., Garg, S., Lee, S., Mullan, P., Sharma, P.: Scalable Grid Service Discovery based on UDDI. Proceedings of the 3rd international workshop on Middleware for grid computing, vol. 117, 1-6, (2005)

Security of Databases and Data

Iosif Schwertner, Krasimira Schwertner
Sofia University, Department of Economics

1 Introduction

Databases (DB) are repositories for important and confidential corporate and private data that should be kept in secure manner. This is the traditional security requirement of the DBs. Nowadays additional regulations govern the security of these data. The most popular regulation is the Sarbanes-Oxley Act (SOX). The Sarbanes-Oxley Act first became law in 2002, shortly after questionable accounting practices were uncovered during the stock market boom of the 1990s -- although most folks will remember 2004 as the first year for mandatory compliance. The purpose of SOX is simple: to protect the public from fraudulent and misleading accounting practices by regulating financial reporting practices. At this point we might be thinking, "I'm in IT -- let the accountants and the CFO deal with SOX compliance." This attitude will definitely get you into serious trouble! At first glance SOX may appear to be an accounting concern, but make no mistake about it -- IT is definitely at the heart of the issue. Most accounting functions rely heavily upon applications that run on computers. Couple this with the fact that financial data is stored in databases, and DBAs in particular will find themselves under close watch and scrutiny for compliance. Sections 302 and 404 of the act have the most relevance for IT governance. In a nutshell, these sections require that all company officers guarantee the accuracy of their financial reports. This in turn requires that all company data used to generate these reports be subject to formal auditing procedures to ensure against incorrect or illegal modifications. Since financial reports rely on the use of a company's IT systems, the security and integrity of these systems is of utmost relevance. It seems logical then that security and change management are, and will continue to be, the most scrutinized aspects of database support. After all, the database is the primary repository for the company's most valued information.

2 Security at file placement level

A database very often keeps sensible data about the users of the database. The data are placed in the files of the database and should be kept in a secure room. But there is no guarantee that the disk drives can't be stolen by an organized group of intruders. So, sensitive data like credit card numbers, addresses, health status, etc. will be misused. The databases are constructed to use many hard disk drives and tablespaces can be placed (and even this is recommended) on different disks. The new modern communication environment often has a very high transmission rate and the link between the different nodes is very fast and reliable. One possibility of keeping data in different places is either to use separated SANs in different physical locations or to divide the data in smaller chunks (if logically possible) or to keep them in different instances (also on different places). The instances can be linked to work together using dblinks. Let us consider stealing of part of the disks. In this case the intruder will be able to use only a part of the data, but it will be impossible or very hard to get all data.

Another way to hide the sensitive data is to make a gateway between the root user data table and the tables with detail data. The idea is to use an intermediate table that hides the primary keys in the main user table introducing a set of different sub keys for different parts of the user data. So normally a user will be represented with a set of random generated identification keys for different tables of the DB. The next step is to encrypt the intermediate table using cryptography tools.

3 Encryption and decryption

Among other security technologies, DBMS protect data in E-business systems through strong, standards-based encryption. E.g. Oracle has supported encryption of network data though Oracle Advanced Security since Oracle7. Oracle9i/10g also supports protection of selected data via encryption within the database.

Although encryption is not a substitute for effective access control, one can obtain an additional measure of security by electively encrypting sensitive data before it is stored in the database. Examples of such data could include:

- credit card numbers
- national identity numbers
- passwords for applications whose users are not database users (passwords are usually hashed with a one way scheme, but it may be necessary to encrypt plain text passwords
- trade secrets
- quarter end profits

To address the need for selective data encryption, DBMS provide packages to encrypt and decrypt stored data. E.g. Oracle provides:

1. The package DBMS_OBFUSCATION_TOOLKIT, supports bulk data encryption using the Data Encryption Standard (DES) algorithm, and includes procedures to encrypt and decrypt using DES.
2. In addition to single DES, DBMS_OBFUSCATION_TOOLKIT supports Triple DES (3DES) encryption, in both two and three key modes, for those who demand the strongest commercial available level of encryption.
3. The Toolkit also supports the MD5 secure cryptographic hash to ensure data integrity.
4. The Toolkit provides a Federal Information Processing Standard (FIPS) 140-certified random number generator for generating secure encryption keys.

Oracle currently supports DES, Triple DES (3DES), AES. They are regarded as the industry standard choice of encryption of sensitive information. The Data Encryption Standard (DES) has been a worldwide encryption standard for over twenty years. The banking industry has also adopted DES-based standards for transactions between private financial institutions, and between financial institutions and private individuals. When industry demands warrants Oracle may decide to incorporate other encryption algorithms such as AES; no plans to support AES are available as of this time.

But Oracle supports AES in his Advanced Security Option, that among other features allows for a broad range of available network encryption algorithms.

3.1 DES Structure

DES is a symmetric key cipher: the same key is used to encrypt data as well as decrypt data. DES encrypts data in 64-bit blocks using a 56-bit key. The DES algorithm ignores 8 bits of the 64-bit key that is supplied. However, developers must supply a 64-bit key to the algorithm.

Triple DES (3DES) is a far stronger cipher than DES because of the longer key length, it can be done in several ways; the most common and secure one is also the one that is implemented by Oracle: encryption; decryption and encryption steps in outer CBC mode. Being the fairly common choice, it allows for inter-operability with other implementations available with third party products.

3.2 DES and Triple DES Security

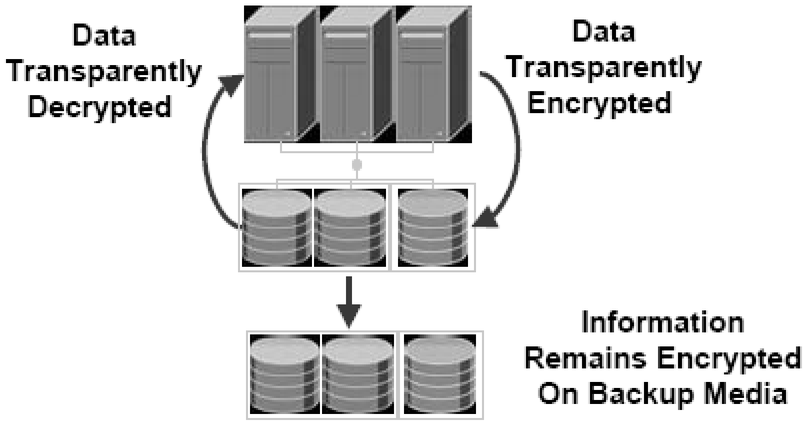
DES has long been the leading encryption method throughout both government and industry for a long time. It is extremely well scrutinized and even the long term objection that some of the rationales behind the detailed structure had not been made public was softened by some recent publications. The key length however remains a point of concern; with the advent of faster CPU's a brute force attack becomes ever more feasible. Hence for the most secure encryption needs, always use Triple DES in three key mode, this has an effective key length of 168 bits.

3.3 ECB and CBC

The DBMS_OBFUSCATION_TOOLKIT supports DES in ECB mode. Triple DES was implemented in CBC mode. With ECB (Electronic Code Book), each 64 bit block is encrypted independently; with CBC (Cipher Block Chaining) a feedback mechanism is added so that every block of encrypted information depends on all previous blocks. This provides an added level of security over ECB since all messages encrypt differently. CBC requires an IV (Initialization Vector) that does not need to be secret but just different for any message that is encrypted with the same key.

4 Transparent Data Encryption (TDE)

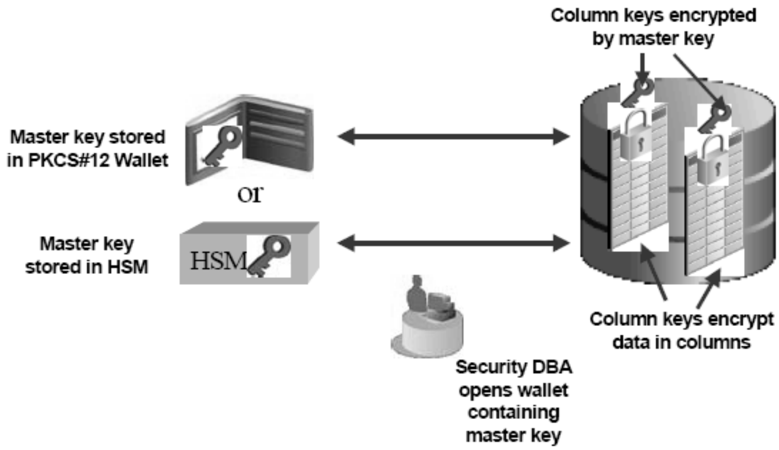
TDE encrypts data before it's written to disk and decrypts data before it is returned to the application. The encryption and decryption process is performed at the SQL layer, completely transparent to applications and users. Subsequent backups of the database files to disk or tape will have the sensitive application data encrypted. Optionally, TDE can be used in conjunction with Oracle RMAN to encrypt the entire Oracle database during backup to disk.


Data Transparently Decrypted
Data Transparently Encrypted
Information Remains Encrypted On Backup Media

Transparent Data Encryption Benefits:

1. Built-in key management
2. Transparent encryption of sensitive application columns
3. Transparent encryption of entire tables paces (New in 11g)
4. Transparent encryption of Secure Files/LOBS (New in 11g)
5. Hardware Security Module (HSM) integration (New in 11g)

Key Management Overview. TDE automatically creates an encryption key when an application table column is encrypted. The encryption key is table specific. If more than one column in a single table is encrypted, the same encryption key is used for all columns. Each table key is stored in the Oracle data dictionary and is encrypted using the TDE master encryption key. The master encryption key is stored outside of the database, in an Oracle Wallet, a PKCS#12 formatted file that is encrypted using a password supplied either by the designated security administrator or DBA during setup. New in Oracle Database 11g Advanced Security is the ability to store the master key in an HSM device using the PKCS#11 interface.


Column keys encrypted by master key
Master key stored in PKCS#12 Wallet
or
Master key stored in HSM
HSM
Column keys encrypt data in columns
Security DBA opens wallet containing master key

Implementation Steps. Since TDE is transparent to existing application code (database triggers and views are not required) the encryption process is simple compared to traditional API based encryption solutions. The following steps can be used to apply TDE:

1. Initialize the master key
2. Identify the sensitive data to encrypt (PII data, credit cards)
3. Verify TDE supports the data type and check foreign key usage
4. Encrypt the sensitive data using TDE

Initialize the Master Key. Master keys are specific to each database. However, any master key can be copied to a secondary database as long as a master key has not been previously established for the secondary database. The master key must be created before any application tables can be encrypted. The syntax to initialize the master key is as follows:

```
SQL> alter system set key identified by “password”;
```

This command creates a wallet and uses the password to encrypt the Wallet, based on the recommendations of the PKCS#5 standard. The Oracle Wallet stores a history of retired master encryption keys and makes them available when data encrypted under an older key is read back from a backup tape.

Identify sensitive data. Identifying PII related data such as social security numbers and credit card can be difficult, especially in a complex application. One technique that can be useful is to search the Oracle data dictionary for column names and data types that are frequently used to store such information.

Check Foreign Key Usage. TDE can't be used to encrypt columns that are used in a foreign key. Verifying whether a column is used as part of a foreign key can be accomplished by examining the Oracle data dictionary.

Encrypting Data Using TDE. To encrypt an existing column:

```
SQL> alter table customers modify (credit_card encrypt);
```

Read consistency is maintained while the encryption transaction is being executed, enabling select (read) operations to continue. DML transaction (insert, update, delete) performed during the encryption transaction will require 'online re-definition'.

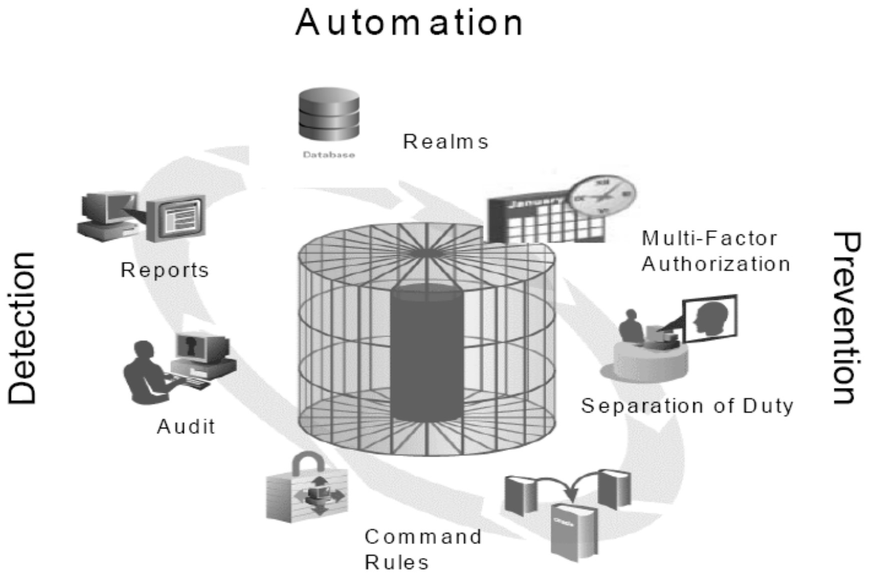
Creating a new table with an encrypted column is easy. The default encryption algorithm is AES192.

```
SQL> create table billing_information (first_name
varchar2(40), last_name varchar2(40), card_number
varchar2(19) encrypt using 'AES256');
```

Transparent Data Encryption works with indexes for equality searches, minimizing the overhead of searching on an encrypted column.

5 Database Vault

Providing accurate and timely access to information, satisfying regulatory compliance requirements, passing internal and external audits, protecting privacy related information, securing applications and databases are just a few of the demands placed on today's IT executive. Traditional approaches to security have relied on the application to enforce access control for application users, the database administrator to maintain both database and application availability and network routers and firewalls to restrict access to the database and application. While problems such as the insider threat are certainly not new, the potential for abuse has never been greater due to the amount of sensitive information being collected and the willingness of criminal organizations and individuals to pay for such information. The nature of the information threat today requires more sophisticated security mechanisms within the database. However, security technologies are efficient only if they can be automated, transparent, provide flexible and adaptable preventions, and provide detection through audit and reporting. Database Vault meets these key requirements for security, providing automation, transparency, prevention and detection through an innovative approach to security similar to the bank vault safe deposit box.



Using realms, Database Vault simplifies the protection of existing applications, preventing access to application data by the DBA or other highly privileged users within the database. Oracle Database Vault introduces multi-factor authorization checks and a rules engine to further protect the database and application. Factor checks and rules enforcement are in addition to existing privileges on access to application data. Oracle Database Vault enforces separation of duty, a critical requirement when applications and databases are consolidated by ensuring that the privileged DBAs are limited to managing the database but not allowed to access the application data. Finally, Oracle Database Vault provides auditing and a comprehensive set of out-of-the-box security reports that provide the detection and reporting component necessary for compliance and auditors.

Blocking Dba Access to Applications. Database administrators play a critical role in maintaining the database. Performance, 24x7 availability and backup/recovery are all part of the DBA job description. These responsibilities place the job of DBA among the most trusted in the enterprise. However, the DBA shouldn't need to access application data residing within the database. The same rule should apply to highly privileged users, such as application owners. These highly privileged users shouldn't be allowed to use their privileges to access application data outside their application. Database Vault realms prevent the DBA, application owners and other privileged users from viewing data outside their job function. At the same time, Database Vault realms allow the DBA and application owner to continue doing their job function. Realms can place a protection boundary around specific tables or an entire application. Realms prevent even the most highly privileged user, including those with the DBA, SYSOPER and SYSDBA, from accessing application data.

Realms are easily defined using the Database Vault administrative web interface. With a few mouse clicks an entire application can be encapsulated within a realm.

Securing Consolidated Databases and Applications. Database consolidation provides tremendous cost savings and more accurate business intelligence. It can also result in increased numbers of DBAs and other highly privileged database users residing in a single database. Database Vault realms provide additional security controls that can prevent unauthorized data access by the DBA or other powerful users in a consolidated database. Database Vault realms allow the benefits of database consolidation to be realized without the security risk.

Internal Threat Prevention. For many years, worms, viruses and the external hacker were perceived as the biggest threats to computer systems. What's often overlooked is the potential for someone who is trusted and with special privileges or access to steal or modify data. Database Vault protects against the insider threat by using realms, factors and command rules. Combined, these provide powerful security to help secure access to databases, applications and sensitive information. Rules and factors can be combined to control the conditions under which commands in the database will be allowed to execute. Rules and factors can be combined to control access to data protected by a realm. For example, factors and rules, like IP Address, time of day and program name can be combined to limit access to only those connections originating from the middle tier during specific operating hours. This can prevent unauthorized access to the application as well as access to the database by unauthorized applications.

Separation of Duty. For security purposes, administration of Database Vault is separate from the day-to-day Oracle DBA function. Database Vault information is protected by a realm that prevents the DBA from attempting to modify realm definitions, factors and command rules defined inside the Database Vault. This separation of duty architecture establishes the strong internal controls necessary to meet various regulatory requirements. Database Vault places additional realms around sensitive objects of the data dictionary owned by SYS and Oracle Label Security to further strengthen security inside the database. Some of the objects protected by Database Vault include roles such as DBA, IMP_FULL_DATABASE, and EXP_FULL_DATABASE. Database Vault realms prevent the DBA role from being granted after Database Vault is installed. In addition, Database Vault requires the Database Vault account manager or someone who has been assigned the dv_acctmgr role to take responsibility for the creation of new accounts in the database. This enforcement overrides all existing accounts with the create user privilege. In other words, even if someone with the create user privilege attempts to create a new database account, Database Vault will block the action. Database Vault installs an Oracle Data Dictionary realm and Account Management realm out-of-the-box to enforce separation of duty. In addition, command rules and factors can be used to further customize operational policies by defining the conditions under which specific database commands can be issued.

References

1. Loney, K., Oracle 9i DBA Handbook, McGraw-Hill/Osborne, 2004
2. Cook, D. R., Database Management From Crisis To Confidence, Oracle Services, Salt Lake City, Utah USA
3. Oracle DataGuard, Concepts and Administration, Release 2 (9.2), October 2002, Part No. A96653-02, Oracle Corporation
4. Database Encryption in Oracle9iR2, An Oracle Technical White Paper, April 2003
5. Oracle Database 10g Release 2 Database Vault, An Oracle White Paper,April 2006

Modified Branch and Bound Algorithm for Solving the Hamiltonian Rural Postman Problem

Andriy Morozov, Anatoliy Panishev

Zhitomir State Technological University, Chernyahovskogo 103, Zhytomyr, Ukraine, 10005
morozov.andriy@gmail.com

Abstract. In this paper the Hamiltonian Rural Postman Problem which is generalization of the Hamiltonian Travelling Salesman Problem is described. The offered modification of a classical method (Little's method) allows to find exact solution of the Hamilton Rural Postman Problem or to detect that the task is unsolvable.

Keywords: Hamiltonian Rural Postman Problem, Hamiltonian cycle, branch and bound algorithm, Hamiltonian Traveling Salesman Problem

1 Introduction

Many outcomes are saved up in research of the Travelling Salesman Problem. They cover the important problems of algorithm design and improvement of methods of combinatorial optimization and their application.

Algorithms with indicators of efficiency which are applicable in real situations aren't known for each applied problem of type of the Travelling Salesman Problem. One of such problems is Rural Postman Problem, formulated as follows [1].

Let *H*=(*V*, *U*) is a connected weighed graph, *V* – set of vertices, $|V| = n$, *U* - set of edges. Each edge $\{i, j\} \in U$ has the weight $d_{ij} \in Z_0^+$, $i \neq j$, $i, j = \overline{1, n}$, Z_0^+ - set of non-negative numbers. The symmetric matrix of weights $\left[d_{ij}\right]_n$ completely defines weighted graph *H*. At this matrix: if $\{i, j\} \in U$ then $d_{ij} \in Z_0^+$, else $d_{ij} = \infty$, $i \neq j$, $i, j = \overline{1, n}$, $d_{ii} = \infty$, $i, j = \overline{1, n}$. The nonempty subset of edges $R \subseteq U$ is given. It is required to find a cycle which includes each edge from *R* and has minimum sum of weights of edges.

Let's designate the Hamiltonian cycle of graph *H* which is passing on all edges of set *R* as $z(R)$. Let's name as the Hamilton Rural Postman Problem (HRPP) the problem which consists in determination of the Hamiltonian cycle $z^*(R)$ minimizing a functional

$$C\left(z(R)\right)=\sum_{\{k,l\}\in z(R)} d_{kl}\,. \tag{2}$$

Interest in solving the HRPP arises when it is required to find an annular route on a transport network of a city or the district, modeled by graph $H=(V,U)$. To each point of departure (arrival) of a network corresponds vertex $i\in V, |V|=n$, and to each edge $\{i,j\}\in U$ corresponds a segment of a road between pair of adjacent points *i* and *j*. Edge $\{i,j\}$ is characterized by weight (cost) d_{ij}. It is equal to expenses for vehicle movement from *i* to *j* or from *j* to *i*.

2 Algorithm

HRPP is NP-complete, because it is a NP-complete Hamiltonian Travelling Salesman Problem (HTSP) in case when $R=\varnothing$.

In [2] there is the algorithm which correctly finds solution of HTSP if graph *H* is Hamiltonian, or detects its unsolvability. Basis of the offered algorithm is a method of branch and bound scheme. It is fulfilled after check of sufficient conditions of unsolvability HRPP. Clearly, that the complexity of such check should be limited by a polynomial from a problem size.

Direct application of branch and bound algorithm from [2] does not allow to solve HRPP. Inclusion of a subset of edges $R\neq\varnothing$ in a required Hamilton cycle turned out so strong restriction that demands other approach to the organization of branching and an evaluation of the lower bound for $C\left(z^{*}(R)\right)$.

Obviously, if graph *H* contains suspended vertices HTSP and HRPP has no solutions. Suspended vertices in graph *H* are finding with complexity $O\left(|V|\right)$. The problems are unsolvable, when graph *H* has a concatenation point [2, 3]. It is required $O\left(|V|+|U|\right)$ elementary operations to define whether graph *H* contains a concatenation point [3].

It is easy to see, that HRPP is unsolvable, if in graph *H* a) the subset of edges of set *R* forms nonhamiltonian cycle; b) there is a vertex, which is incident three or more edges from *R*. Therefore, graph *H* in which the set of edges *R* does not form a collection of vertex-not crossed chains, does not contain cycle $z(R)$. Time limited in size $O\left(|V|+|U|\right)$ suffice to check the conditions.

It is known from [4], that HRPP can be transformed to the same task on graph $H=(V,U)$ in which a) degrees of all vertices are higher 2, b) the set of edges *R*, which is contain in a required Hamiltonian cycle, organizes matching *R*.

Let's notice, that $|R|$ limits a solution space so, that it can be empty. Let's name admissible solution $z(R)$ of HRPP by tour.

Search the solution of the task starts with conversion a matrix of costs of graph H to the reduced matrix [5]. The reason is that there is minimum element $\alpha_i = \min_j d_{ij}$, $i = \overline{1,n}$ in line i which is subtracted from each element of this line. Then in a column j minimum element $\beta_j = \min d_{ij}, j = \overline{1,n}$ which is subtracted from each element of this column is retrieved. Elements α_i and β_j are named as reducing coefficients. From the reduced matrix $\left[d_{ij}\right]_n$ the lower bound of cost of the required solution is searched:

$$\varphi(R) = \sum_{\{i,j\}\in R} \min\left\{d_{ij}, d_{ji}\right\} + \gamma , \qquad \textbf{(3)}$$

where $\gamma = \sum_{i=1}^{n} \alpha_i + \sum_{j=1}^{n} \beta_i$.

Generally, the reduced matrix which defines set of all solutions HRPP is not symmetric. To this matrix the weighed multigraph $H' = (V, U')$, in which vertices i and j are linked by pair arcs (i, j) and (j, i) if in graph $H = (V, U)$ they are linked by edge $\{i, j\}$, corresponds one-to-one. Thus, each edge $\{i, j\} \in R$ of graph H is presented in multigraph H' by two arcs $(i, j) \in R'$, $(j, i) \in R'$, $|U'| = 2|U|$, $|R'| = 2|R|$.

From the reduced matrix we select elements (i, j) and $(j, i) \in R'$ for which $\Delta_{ij} = \min\left\{d_{ij}, d_{ji}\right\} > 0$ and let
$d_{ij} = d_{ij} - \min\left\{d_{ij}, d_{ji}\right\} = d_{ij} - \Delta_{ij}$, $d_{ji} = d_{jji} - \min\left\{d_{ij}, d_{ji}\right\} = d_{ji} - \Delta_{ij}$.

Let's name obtained matrix $\left[d_{ij}\right]_n$ completely reduced. The further operations will be fulfilled by means of this matrix.

Conversion of a matrix of costs of graph H to completely reduced matrix has the argued substantiation. HRPP for completely reduced matrix consists in construction the Hamiltonian cycle or detour the minimum cost which includes exactly one arc from each pair (i, j), $(j, i) \in R'$, containing at least one arc with zero weight. Thus, the list of arcs of the zero weight which are candidates for inclusion in optimal solution is defined on matrix $\left[d_{ij}\right]_n$.

The explained reasons open possibility to adapt the classical algorithm of branch and bound described in [6], for searching the solution of the HRPP. The mode of branching of admissible solutions $z(R)$ keeps within the known scheme of construction of a binary search tree [6].

Let's put in correspondence to the root of a tree of search $\{z(R)\}$ completely reduced matrix $\left[d_{ij}\right]_n$ with bound $\varphi(R)$, define the arc of multigraph H' which initiates branching. To realize it, as well as at [6], each element (p,q) in $\left[d_{ij}\right]_n$, if $d_{pq}=0$, let's estimate as value

$$\gamma(p,q)=\min_{i\neq q} d_{pi}+\min_{j\neq p} d_{jq}, \tag{4}$$

also find the element (p,q) which has the greatest value

$$\Gamma(k,l)=\max\left\{\gamma(p,q)\,|\,d_{pq}=0\right\}, \tag{5}$$

In multigraph H', the arc (p,q) corresponds to element (p,q). This arc initiates a partition of set of all detours to two subsets and, in the conditions of HRPP, generates two cases: $\{k,l\}\notin R$ and $\{k,l\}\in R$.

In case of $\{k,l\}\notin R$ the set of all solutions of the problem is divided into subsets $\{\{k,l\}\notin R\}$ and $\{(\overline{k,l})\}$. The first subset includes all detours which contain arc (k,l), the another one includes all detours which do not contain this arc.

The matrix by which the low bound $\varphi((k,l)\in R')$ of cost of all detours of set $\{\{k,l\}\notin R'\}$ is calculated, is provided as in [6] according to the following rule.

If the set *R* contains the edge $\{x,k\}$, the required detour together with the arc (k,l) joins the arc (x,k). Similarly, if the set *R* contains the edge $\{y,l\}$, the arc (k,l) joins the arc (l,y). Inclusion of the arc (x,k) or the arc (l,y) in a subset of solutions $\{(k,l)\notin R'\}$ means, that the matrix which defines it, does not contain the line *k* and the column *l*, as well as the line and the column, numbers of which are the beginning and the end of the joined arc. In case of $\{x,k\}$, $\{y,l\}\in R$, the arcs (x,k) and (l,y) join to arc (k,l), and in a matrix, which defines the set $\{(k,l)\notin R'\}$, the lines *x*, *k*, *l* and the columns *k*, *l*, *y* exclude.

Let's designate the lower bound of branching vertex as φ. For an arc $(k,l)\notin R'$ which initiates branching, let

$\mu_k = \begin{cases} d_{xk}, \text{ if } \{x,k\} \in R, \\ 0, \text{ else.} \end{cases}$, $\mu_l = \begin{cases} d_{ly}, \text{ if } \{l,y\} \in R, \\ 0, \text{ else.} \end{cases}$

Here d_{xk}, d_{ly} - the elements of a matrix which corresponds to the bound φ. In case $\varphi = \varphi(R)$ they are the elements of the completely reduced matrix $\left[d_{ij}\right]_n$. Then, the cost of all detours of the set $\{(k,l) \notin R'\}$ is bounded below by the value

$$\varphi((k,l) \notin R) = \varphi + \mu_k + \mu_l + \sum \alpha_i' + \sum \beta_j' , \qquad \textbf{(6)}$$

where α_i' and β_j' - the reduction factors. Those coefficients are obtained as a result of transformation of a matrix, which to the bound φ corresponds, to a matrix, which defines the set $\{(k,l) \notin R\}$.

The matrix which defines a set $(\overline{k,l})$, and the bound $\varphi(\overline{k,l})$ are presented by the value $\varphi(\overline{k,l}) = \varphi + S(k,l)$, where $S(k,l)$ is the sum of reduction factors which is obtained as a result of reduction of a matrix of costs.

Let's describe a case $(k,l) \in R$. Solution space HRPP $\{z(R)\}$ is divided into two subsets $\{(k,l)\}$ and $\{(l,k)\}$. The first subset contains all detours which include the arc (k,l), the second one - all detours which include the arc (l,k).

To the branching vertex boundary φ and matrix *D* which generates arc (k,l) or (l,k), $\{k,l\} \in R$, with a maximum estimation let correspond. The matrix which defines subset $\{(k,l)\}$, is a result of elimination from *D* a line *k* and a column *l*, appropriations the value ∞ to element d_{lk} and reduction of the obtained shorted matrix. Similarly, for determination a matrix which defines subset $\{(l,k)\}$, it is necessary in *D* to suppose $d_{kl} = \infty$, to remove a line *l* and a column *k* and to fulfill the reduction of the obtained shorted matrix. The lower boundaries of cost of detours for subset $\{(k,l)\}$ and $\{(l,k)\}$ are calculated as follows:

$$\varphi(k,l) = \varphi + d_{kl} + \sum_{i \neq k} \alpha_i' + \sum_{j \neq l} \beta_j' , \qquad \textbf{(7)}$$

$$\varphi(l,k) = \varphi + d_{lk} + \sum_{i \neq k} \alpha_i' + \sum_{j \neq l} \beta_j' , \qquad \textbf{(8)}$$

where α_i', β_j' - reduction factors which are obtained as a result of transformation of a matrix *D*.

The situation when either $d_{kl} = \infty$, or $d_{lk} = \infty$ in D is possible. It follows from (1), that if $d_{kl} = \infty$, $\{(k,l)\} = \varnothing$, if $d_{lk} = \infty$, $\{(l,k)\} = \varnothing$.

Modified Little's method of solving the Hamiltonian Rural Postman Problem has the following appearance.

0. $H = (V,U)$ - the nonoriented weighed graph with degrees of vertices higher than 2, presented by a matrix of costs with power $n = |V|$; R - a matching of graph *H* which contains in required Hamiltonian cycle $z^*(R)$ of the minimum cost. Let $C^* = \infty$.

1. The matrix of costs of graph *H* is transformed to the reduced matrix. From it, the lower bound $\varphi(R)$ of costs for all solutions of a problem is calculated. From the reduced matrix completely reduced matrix $D = \left[d_{ij}\right]_n$ which corresponds to the oriented weighed multigraph $H' = (V,U')$, $|U'| = 2|U|$, $|R'| = 2|R|$ is defined. In H' it is required to find the Hamiltonian cycle (detour) which contains exactly the one arc from each pair of arcs (i,j), $(j,i) \in R'$ and has the minimum cost.

2. From matrix *D* under formulas (3) and (4) an element (k,l) is calculated. In a multigraph it is presented by an arc (k,l) which generates the branching vertex in a search tree.

3. If $\{k,l\} \notin R$, then go to step 7.

4. If $d_{lk} = \infty$, then a subset of detour $\{(l,k)\}, \{k,l\} \in R$, which contains an arc (l,k) is empty; go to step 6.

5. To define a subset of detours $\{(l,k)\}, \{k,l\} \in R$ which contain an arc (l,k), the reduced matrix in which $d_{kl} = \infty$ and with excluded line *l* and a column *k* is considered. After matrix reduction under the formula (7), the lower boundary $\varphi(l,k)$ of cost of arcs of all detours of subset $\{(l,k)\}$ is calculated. Subset $\{(l,k)\}$ generates in a search tree the current active vertex $\{(l,k)\}$. If $C^* > \varphi(l,k)$ then vertex $\{(l,k)\}$ with reduced matrix *D* and bound $\varphi(l,k)$ joins branching vertex, else it is not considered further. Go to step 9.

6. To determinate a subset of detours $\{(k,l)\}, \{k,l\} \in R$ which contain an arc (k,l), the reduced matrix in which $d_{lk} = \infty$ and both a line *k* and a column *l* are excluded is considered. Under the formula (6) the lower bound $\varphi(k,l)$ of cost of all detours of subset $\{(k,l)\}$ which initiates in a searching tree the current active vertex

$\{(k,l)\}$ is calculated. If $C^* > \varphi(k,l)$, then a vertex $\{(k,l)\}$ with a reduced matrix *D* and a bound $\varphi(k,l)$ joins the branching vertex, otherwise the vertex is excluded from the further reviewing; go to step 9.

7. In determination of a subset of detours $\{(\overline{k,l})\}, \{k,l\} \notin R$ which is not containing arcs (k,l), the matrix in branching vertex is reduced to matrix *D* after replacement $d_{kl} \neq \infty$ on $d_{kl} = \infty$. If $C^* > \varphi(\overline{k,l})$ then in a searching tree a vertex $\{(\overline{k,l})\}$ is added to the branching vertex together with a reduced matrix *D* and a lower bound $\varphi(\overline{k,l})$ of cost of detours $\{(\overline{k,l})\}$, otherwise it has no further prolongation. The bound $\varphi(\overline{k,l})$ is equal to the lower bound of vertex of branching increased by the sum of reduction factors.

8. For determination of a subset of detours $\{(k,l) \notin R'\}$ which contain arc (k,l), in a matrix at branching vertex the line *k* and a column *l* is excluded; lines *s*, *k* and columns *k*, *l*, if $\{s,k\} \in R$, are excluded; lines *k*, *l* and columns *p*, *l*, if $\{l,p\} \in R$, are excluded; lines *s*, *k*, *l* and columns *k*, *l*, *p*, if $\{s,k\},\{l,p\} \in R$, are excluded. Matrix *D* is searched by reduction of the obtained matrix. After an evaluation of the lower bound $\varphi((k,l) \notin R')$ of cost of detours $\{(k,l) \notin R'\}$ which is defined under formulas (5), a vertex $\{(k,l) \notin R'\}$ with a matrix D and a bound $\varphi((k,l) \notin R')$ at performance of condition $C^* > \varphi((k,l) \notin R')$ joins branching vertex, otherwise it is excluded from review.

9. If dimension of matrix *D* is equal to 1, admissible solution of HRPP is obtained. If the estimation of vertex of a branching tree is less than $C(z(R))$, it is remembered obtained admissible solution $z(R)$. We appropriate an estimation of current vertex as $C(z(R))$. We exclude from the further reviewing the vertices which have estimation more or equal C^*. If there is no remained vertex after elimination, go to step 10. Otherwise, it's necessary to find a vertex of searching tree with the minimal estimation. If there are such vertices, chose vertex which has the greatest depth.

10. If no admissible solution is found, then HRPP is unsoluble. Otherwise $z^*(R)$ - a required Hamiltonian cycle, and C^* - its weight.

The method has been implemented in C# programming language. For performance tests we have used Intel Core 2 Duo 1,6GHz/2Gb RAM PC. Time of solving the Travelling Salesman Problem for dimension 50 occupies an average of 5 minutes.

3 Example

Graph $H = (V, U)$ is defined by a matrix of costs

$$\left[d_{ij}\right]_6 = \begin{array}{c|cccccc|} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & \infty & 10 & \infty & 5 & 7 & 18 \\ 2 & 10 & \infty & 23 & 15 & \infty & 8 \\ 3 & \infty & 23 & \infty & 25 & 24 & \infty \\ 4 & 5 & 15 & 25 & \infty & \infty & 23 \\ 5 & 7 & \infty & 24 & \infty & \infty & 11 \\ 6 & 18 & 8 & \infty & 23 & 11 & \infty \end{array}.$$

Subset of edges $R = \{\{3,4\},\{1,5\},\{2,6\}\}$. It is required to find a solution of HRPP or to detect that it has no solution.

1. We reduce given matrix of costs and obtain reduced matrix

$$\left[d_{ij}\right]_6 = \begin{array}{c|cccccc|} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & \infty & 5 & \infty & 0 & 1 & 13 \\ 2 & 2 & \infty & 0 & 7 & \infty & 0 \\ 3 & \infty & 0 & \infty & 2 & 0 & \infty \\ 4 & 0 & 10 & 5 & \infty & \infty & 18 \\ 5 & 0 & \infty & 2 & \infty & \infty & 4 \\ 6 & 10 & 0 & \infty & 15 & 2 & \infty \end{array}.$$

We calculate the lower bound of costs of all HRPP solutions:

$$\varphi(R) = \min\{d_{34}, d_{43}\} + \min\{d_{15}, d_{51}\} + \min\{d_{26}, d_{62}\} + \sum_{i=1}^{6}\alpha_i + \sum_{j=1}^{6}\beta_j =$$

$$= \min\{2,5\} + \min\{1,0\} + \min\{0,0\} + 56 + 16 = 74.$$

$\Delta_{34} = min\{d_{34}, d_{43}\} = d_{34} = 2 > 0$, therefore let $d_{34} = 0$, $d_{43} = 3$, we obtain completely reduced matrix

$$\left[d_{ij}\right]_6 = \begin{array}{c|cccccc|} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & \infty & 5 & \infty & 0 & 1 & 13 \\ 2 & 2 & \infty & 0 & 7 & \infty & 0 \\ 3 & \infty & 0 & \infty & 0 & 0 & \infty \\ 4 & 0 & 10 & 3 & \infty & \infty & 18 \\ 5 & 0 & \infty & 2 & \infty & \infty & 4 \\ 6 & 10 & 0 & \infty & 15 & 2 & \infty \end{array}.$$

2. We calculate estimations for each zero element of completely reduced matrix: $\gamma(1,4) = 1, \gamma(2,3) = 2, \gamma(2,6) = 4, \gamma(3,2) = 0, \gamma(3,4) = 0, \gamma(3,5) = 1, \gamma(4,1) = 3,$ $\gamma(5,1) = 2, \gamma(6,2) = 2$. Arc $(2,6)$ of multigraph H' initiates branching.

3. $\{2,6\} \in R$.

4. $d_{26} \neq \infty$. The set of all solutions is divided into a subset of detours $\{(6,2)\}$, which contains arc $(6,2)$ and a subset of detours $\{(2,6)\}$, which contains arc $(2,6)$.

5. We calculate a matrix which defines a subset of detours $\{(6,2)\}$. We exclude the arc $(2,6)$, let $d_{26} = \infty$, also we exclude line 6 and column 2 . We reduce the obtained matrix and obtain new matrix

$$\left[d_{ij}\right]_5 = \begin{array}{c|ccccc} & 1 & 3 & 4 & 5 & 6 \\ \hline 1 & \infty & \infty & 0 & 1 & 9 \\ 2 & 2 & 0 & 7 & \infty & \boxed{\infty} \\ 3 & \infty & \infty & 0 & 0 & \infty \\ 4 & 0 & 3 & \infty & \infty & 14 \\ 5 & 0 & 2 & \infty & \infty & 0 \end{array}.$$

This matrix gives estimation

$$\varphi(6,2) = \varphi(R) + d_{62} + \sum_{i \neq 6} \alpha_i' + \sum_{j \neq 2} \beta_j' = 74 + 0 + 4 = 78.$$

6. We obtain a matrix which defines a subset of detours $\{(2,6)\}$. To the arc $(6,2)$ we assign the weight $d_{62} = \infty$, exclude line and column 6. We reduce the obtained matrix:

$$\left[d_{ij}\right]_5 = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 5 & \infty & 0 & 1 \\ 3 & \infty & 0 & \infty & 0 & 0 \\ 4 & 0 & 10 & 1 & \infty & \infty \\ 5 & 0 & \infty & 0 & \infty & \infty \\ 6 & 8 & \boxed{\infty} & \infty & 13 & 0 \end{array}.$$

We find an estimation:

$$\varphi(2,6) = \varphi(R) + d_{26} + \sum_{i \neq 2} \alpha_i' + \sum_{j \neq 6} \beta_j' = 74 + 0 + 2 + 2 = 78.$$

9. Dimension of the reduced matrix is more than 1. Any of vertices $\{(2,6)\}$, $\{(6,2)\}$ of branching tree can be chosen as the active. We choose the vertex $\{(2,6)\}$.

2. We calculate estimations for each zero element of the matrix: $\gamma(1,4) = 1$, $\gamma(3,2) = 5, \gamma(3,4) = 0, \gamma(3,5) = 0, \gamma(4,1) = 1, \gamma(5,1) = 0, \gamma(5,3) = 1$, $\gamma(6,5) = 8$. Arc $(6,5)$ initiates branching.

3. $\{6,5\} \notin R$; the set of detours $\{(2,6)\}$ is divided into a subset of detours $\{(6,5) \notin R'\}$ which includes arc $(6,5)$, and a subset of detours $\{(\overline{6,5})\}$ which does not contain this arc.

7. In the matrix which defines a subset of detours $\{(2,6)\}$, let $d_{65}=\infty$ and reduce obtained matrix. Outcome of such operations is the matrix

$$\left[d_{ij}\right]_5=\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 5 & \infty & 0 & 1 \\ 3 & \infty & 0 & \infty & 0 & 0 \\ 4 & 0 & 10 & 1 & \infty & \infty \\ 5 & 0 & \infty & 0 & \infty & \infty \\ 6 & 0 & \infty & \infty & 5 & \boxed{\infty} \end{array}$$

and estimation $\varphi\left(\overline{6,5}\right)=\varphi(2,6)+S(6,5)=78+8=86$ which limits from below costs of all detours $\left\{\left(\overline{6,5}\right)\right\}$. Set $\left\{\left(\overline{6,5}\right)\right\}$ includes arc $(2,6)$ and does not contain arc $(6,5)$.

8. In a matrix which corresponds to the lower bound $\varphi(2,6)$, we exclude line 6 and column 5. As edges $\{6,5\}\notin R$ and $\{5,1\}\in R$ are adjacent, we exclude line 5 and column 1 and we forbid arc $(1,2)$ assigning $d_{12}=\infty$ to exclude subcycles. We obtain a matrix

$$\left[d_{ij}\right]_3=\begin{array}{c|ccc} & 2 & 3 & 4 \\ \hline 1 & \boxed{\infty} & \infty & 0 \\ 3 & 0 & \infty & 0 \\ 4 & 9 & 0 & \infty \end{array}$$

and estimation $\varphi\left((6,5)\notin R'\right)=\varphi(2,6)+d_{51}+\sum\alpha_i'+\sum\beta_j'=78+0+1=79$.

9. Dimension of this matrix is more than 1, we select the vertex $\varphi(6,2)$ which has the minimal lower estimation.

2. We calculate an estimation for each zero element:
$\gamma(1,4)=1,\gamma(2,3)=4,\gamma(3,4)=0,\gamma(3,5)=1,\gamma(4,1)=5,\gamma(5,1)=0,\gamma(5,6)=9$.
The further branching is fulfilled by means of an arc $(5,6)$.

3. $\{5,6\}\notin R$, the set of detours $\{(6,2)\}$ is presented by a division into subset $\{(5,6)\notin R'\}$ and subset $\left\{\left(\overline{5,6}\right)\right\}$.

7. To vertex $\left\{\left(\overline{5,6}\right)\right\}$ corresponds the matrix

$$\left[d_{ij}\right]_5=\begin{array}{c|ccccc} & 1 & 3 & 4 & 5 & 6 \\ \hline 1 & \infty & \infty & 0 & 1 & 9 \\ 2 & 2 & 0 & 7 & \infty & \infty \\ 3 & \infty & \infty & 0 & 0 & \infty \\ 4 & 0 & 3 & \infty & \infty & 14 \\ 5 & 0 & 2 & \infty & \infty & \boxed{\infty} \end{array}$$

and an estimation $\varphi\left(\overline{5,6}\right)=\varphi(6,2)+S(5,6)=78+9=87$.

8. In the matrix which describe set of detours $\{(5,6)\notin R'\}$, we assign $d_{21}=\infty$ to exclude subcycles:

$$\left[d_{ij}\right]_3=\begin{array}{c|ccc} & 1 & 3 & 4\\ 2 & \boxed{\infty} & 0 & 7\\ 3 & \infty & \infty & 0\\ 4 & 0 & 3 & \infty\end{array}.$$

We calculate an estimation of vertex:
$\varphi\left((5,6)\notin R'\right)=\varphi(6,2)+d_{15}=78+1=79$.

9. For branching we choose vertex $\{(6,5)\notin R'\}$ with the minimal lower bound.

2. We calculate estimations for each zero element of the matrix which corresponds to current vertex:
$\gamma(1,4)=0,\gamma(3,2)=9,\gamma(3,4)=0,\gamma(4,3)=9$.
Branching is continued by an arc $(3,2)\notin R'$.

7. Matrix

$$\left[d_{ij}\right]_3=\begin{array}{c|ccc} & 2 & 3 & 4\\ 1 & \infty & \infty & 0\\ 3 & \boxed{\infty} & \infty & 0\\ 4 & 0 & 0 & \infty\end{array}$$

defines set of detours $\left\{\left(\overline{3,2}\right)\right\}$ with lower bound $\varphi\left(\overline{3,2}\right)=\varphi(6,5)+S(3,2)=$
$=79+9=88$

8. In a matrix which corresponds to vertex $\left\{\left(\overline{6,5}\right)\notin R'\right\}$, we assign $d_{32}=\infty$ and exclude lines 3, 4 and columns 2, 3.

9. As a result we obtain a matrix of the one element (1, 4) and, $\{(2,6)\}$. Therefore we obtain detour (2,6), (6,5), (5,1), (1,4), (4,3), (3,2) with cost $\varphi\left((3,2)\notin R'\right)=\varphi\left(\left(\overline{6,5}\right)\notin R'\right)+d_{43}=79+0=79$.

All vertices of the branching tree have estimations, greater or equal 79, go to step 10.

10. The obtained admissible solution of problem
$z(R)=\{(2,6),(6,5),(5,1),(1,4),(4,3),(3,2)\}$ with cost $C(z(R))=79$ is optimum.

The Fig. 1 illustrates the solution tree of this example.

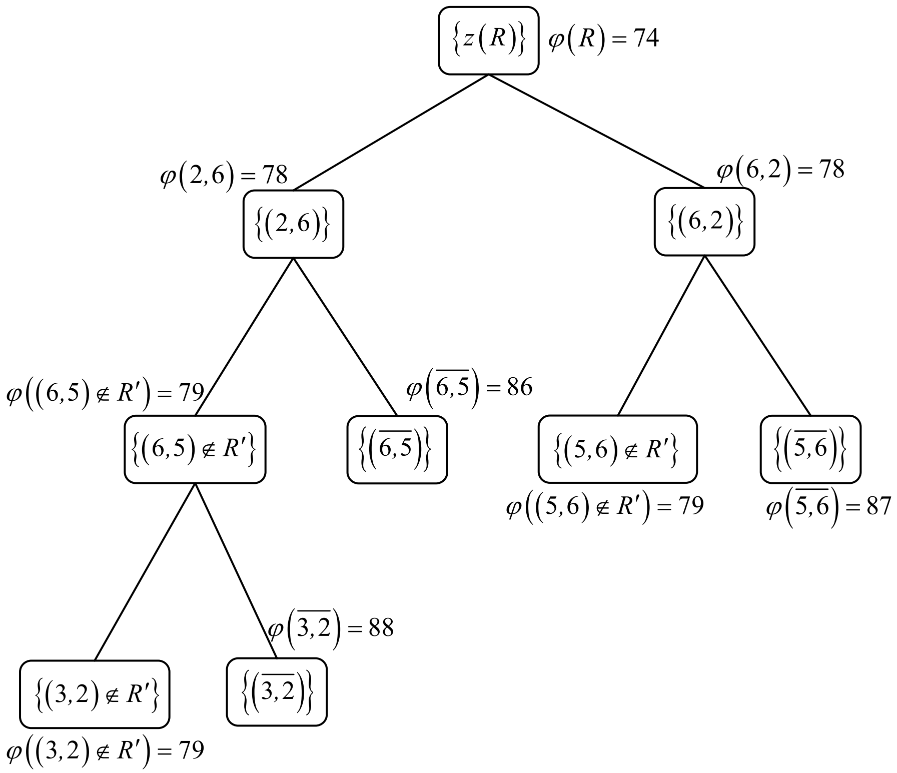


Fig. 11. The Solution Tree

References

1. Gary M., Jonson D.: Computing machines and intractable problems/Вычислительные машины и труднорешаемые задачи. Mir, Moscow (1982)
2. Garashchenko I., Panishev A.: Method of Finding Hamilton Routes in Transport Network. Artificial Intelligence and Decision Making, number 7, pp. 43-48, ITHEA, Sofia (2008)
3. Koffman E. (edited by): The theory of schedules and computing machines/Теория расписаний и вычислительные машины. Nauka, Moscow (1984)
4. Garashchenko I., Morozov A., Panishev A.: The Method of Solving of the Hamiltonian Travelling Salesman Problem/Метод решения гамильтоновой задачи коммивояжера. Artificial intelligence, volume 3, pp. 630-637 (2008)
5. Yablonsky A.: Minimisation of cyclic paths of delivery of production to consumers/ Минимизация кольцевых маршрутов доставки продукции потребителям. Economy and mathematical methods, vol. 43, №3, pp. 124-131 (2006)

Service Functionality for Cloud Systems [6]

Radko Zhelev, Vasil Georgiev

Institute for Parallel Processing – Bulgarian Academy of Sciences
zhelev@acad.bg

Abstract. This paper discusses the concept of Cloud Systems' organization and the variety of service functionality extents. Its main purpose is to outline the general definition and structure of a Cloud System, as well as to illustrate the system boundaries and aspects of Cloud Service functionality. The study analyzes all levels of the Cloud architecture, explores concrete realizations and classifies how they fit to the general architecture.

Keywords: Cloud Computing, IAAS, PAAS, SAAS, SOA, Virtualization, Scalability.

1. Introduction

Cloud Computing technology gains much popularity and growing attraction in the IT environment. Since the technology perfectly fits to the real-life atmosphere providing excellent benefit to the Cloud System consumers, expressed in time and money savings, the Cloud Computing sphere becomes a very business attractive target and main objective for the companies to focus on. The biggest IT companies are starting to build their Cloud Infrastructures employing their big capacity of specialists, ready-to-use technologies as well as existing experience in virtualizations, operating systems, distributed computing and scaling. They are hurrying to quickly prepare their systems and provide attractive extent of Cloud Services, in order to occupy a good place in the market competition. This imposes that business researches outstrip the scientific educational progress in this sphere, and thus - most of the papers and readings available on this topic are business-oriented company-specific presentations focusing on their own solutions, which in turn contain quite much advertising information along with the technical descriptions on the general concept and system organization. This makes it difficult to trace all definitions, and summarize a general concept and overall idea about the system boundaries.
This paper tries to outline the system boundaries and aspects of Cloud Service functionality.
The paper proceeds as follows. First, in Section 2 is provided a general definition and architecture of a Cloud System. Section 3 explores existing Cloud Systems and classifies how they fit to the general architecture. Shows how realizations of the same service level may provide different service extent, and figures pros and cons for each strategy. Finally, the conclusion chapter presented in Section 4 provides some

[6] This paper is supported by the project FMI_Grid 062/15.05.2009

comparative analysis between the new Cloud technology and the familiar Grid Systems.

2. Cloud – definition and general characteristics

2.1 General definition of a Cloud System

A Cloud System could be determined as - system in which dynamically scalable and often virtualized resources are provided as a service over the Internet. Users need not have knowledge of, expertise in, or control over the technology infrastructure "in the cloud" that supports them [1].
The concept involves several levels of provided services – IAAS, PAAS and SAAS.

Infrastructure As A Service (IAAS)
This is the providing of Computing Infrastructure as a service. Consumers can rent virtualized hardware resources – servers, network equipment, memory, CPU power, disk storage equipped with basic software like operating systems for controlling those resources. For instance, a customer can rent arbitrary number of servers with certain hardware configurations - CPU power, RAM and disk storage. The Service allows dynamic reservation and releasing of resources, thus facilitating the scalability and eliminating the need for preliminary capacity planning by the consumers. Consumers can benefit from using a prepared infrastructure without the need to buy and maintain certain hardware and employ system administrators. Meanwhile they wont have to use more powerful (and thus - underutilized) infrastructure than actual needs require, just in case the demand starts to grow. With the flexibility of dynamic reservation, they can rent less hardware resources and easily extend the employed infrastructure when and if the needs require. On behalf of the Cloud, the virtualization of the underlying resources allows a number of hardware "tenants" to exist together absolutely isolated on a single physical resource, thus increasing the hardware utilization.

Platform As A Service
This could be the so called Middleware, representing a software platform for development and integration of complete software applications in a Cloud Infrastructure. The platform could be targeted for hosting standard applications – like web, or could be specialized around a particular area. It could provide general services applicable for software platforms – logging systems, database systems, web portal tools, etc. The underlying realization of all should be in relevance with the Cloud System paradigm – since implemented on the top of a large distributed infrastructure, they should be realized with adequate load-balancing, scalability and reliability. It is up to the platform concept to define whether to hide the whole distribution handling inside "the cloud", or to export distribution-related services relative to the underlying infrastructure – for instance communication protocols or

load distribution algorithms that could be used by more sophisticated applications enabled for horizontal scaling to handle the control over the load by themselves.

Software As A Service

These are complete software applications provided as services for usage by other applications or end customers. The difference in using an application as a service is that the applications are accessed remotely on demand by the client, without the need to keep a local copy of the application. This allows single application instances to serve multiple clients. Additional benefit to the consumers is the new license agreements allowing the customer to pay for the service as much as they really use it, without the need to buy complete licenses for different software applications.

Summary of the Cloud System goals

Instead of doing expenses for software licenses and hardware equipment to build own infrastructures, consumers can use everything as a completely external service (just like the electricity consumption) and pay precisely for what they use, based on the hardware resource characteristics, time usage, i/o traffic, etc. Such a service could bring the benefits of saving the purchase and maintenance of hardware, installing and configuring the environment, employment and education of qualified staff, system capacity planning, complications in extending the ready environment when the needs for productivity grow.

2.2 Cloud System architecture

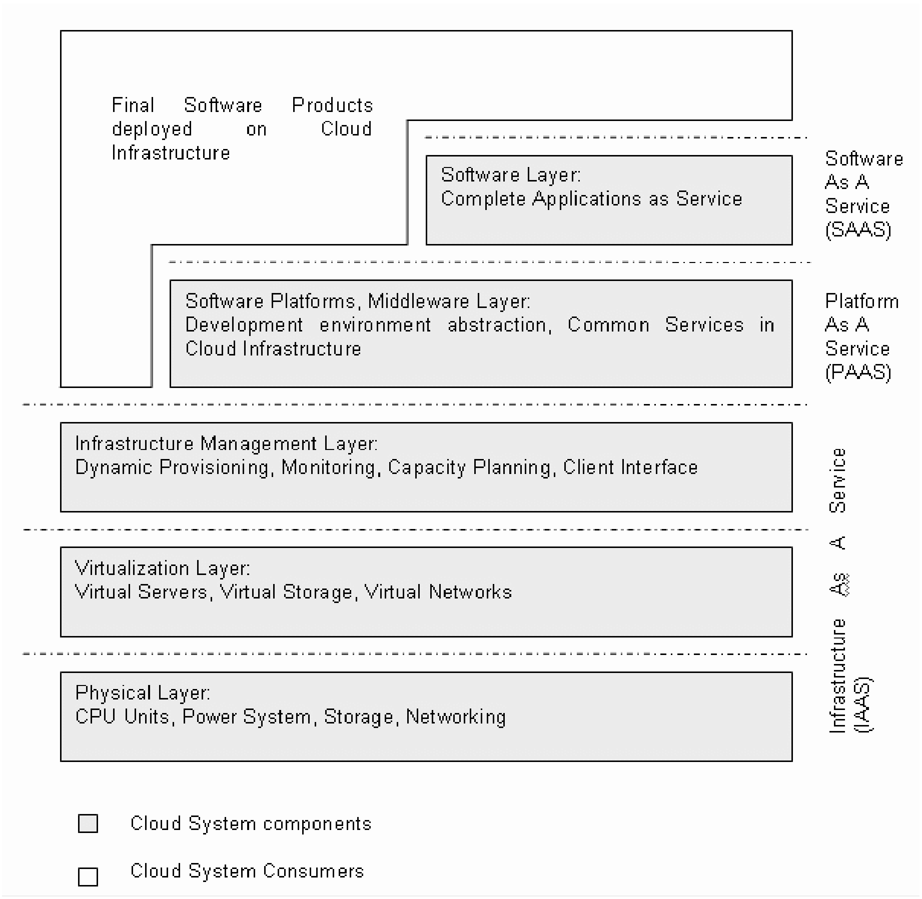

Final Software Products deployed on Cloud Infrastructure
Software Layer: Complete Applications as Service
Software As A Service (SAAS)
Software Platforms, Middleware Layer: Development environment abstraction, Common Services in Cloud Infrastructure
Platform As A Service (PAAS)
Infrastructure Management Layer: Dynamic Provisioning, Monitoring, Capacity Planning, Client Interface
Virtualization Layer: Virtual Servers, Virtual Storage, Virtual Networks
Physical Layer: CPU Units, Power System, Storage, Networking
Infrastructure As A Service (IAAS)
Cloud System components
Cloud System Consumers

Fig. 12. Cloud System Architecture

Physical Layer

In the bottom of a Cloud Infrastructure stays the physical hardware – computing resources on the top of which the system is built. In the general case these are 'farms' of computers, hard drives, CPU units, power supplies, coolers, network equipment, etc

Virtualization Layer

Virtualization is a technique for abstracting of logical resource over underlying physical resource for gaining more flexibility and utilization of the physical resource. More often – few logical resources share independently the same physical one. In the virtualized environment, different computing environments may dynamically be created, extended, replaced as the requirements vary.

Virtualization fits well to the dynamic ideology of Cloud, since it brings key benefits like resource sharing, convenient management and isolation. Virtualization allows a number of underutilized physical servers to be logically consolidated into a smaller number of more fully utilized physical servers.

There are two implementation types of virtualization:

- **on hardware level** (firmware based) – the virtualization directly uses the hardware primitives. Hardware itself should support logical separation, for instance LPAR (term defined by IBM – Logical Partitions [2], [3]) – subset of hardware resources virtualized as a separate computer.
- **on software level** – virtualization is realized on the top of host operating system (software based), using the operating system methods for accessing the hardware. The virtualization program is called VM (Virtual Machine) – software implementation of computer, which executes program as a real machine [4]. Examples: VMware Server, Microsoft Virtual Server.

The hardware virtualization is more efficient, since it accesses directly the hardware, while the software based is more general and convenient for universal usage in heterogeneous systems.

In this layer, the resources from the physical layer are virtualized and represented as logical resources that can be created, destroyed, extended, replaced, reassembled with arbitrary logical components, hiding the underlying static devices.

Infrastructure Management Layer

Virtualization is a key technique for the realization of the IAAS service provided by a Cloud System. But achieving a real efficiency in resource utilization requires a management layer that controls and organizes the effective usage of the resources in the global environment [2]. The responsibilities of this layer are to organize:

- **automatic provisioning** of the virtualized resources.

It must be possible the rented resources to be automatically provisioned with operating systems, to be initially configured, supplied with additional basic software, configured with network storage, etc. All these operations are time-consuming and error-prone if being done manually, and on the other hand – they are really frequently performed in the context of a Cloud System. That's why the automatic provisioning is a key feature for the providing the resources as a service.

- **monitoring, statistics, capacity planning, automation**

The Cloud System must monitor the status of availability and configuration of system resources, to schedule occupation and releasing of resources, as well as to reserve resources for future use. Critical point for the management of computer resources is the ability to estimate the current and future capacity of the system, in order to accept new queries. Without this estimation, it can not be planned how much clients can be supported, as well as a stable pipeline of applications to be assured.

- **client interface**

Responsibility of this layer is to interact with the clients of the IAAS Service, so it must provide a client interface (user interface or program interface) defining means for accessing and operating the service by the consumers.

Software Platforms, Middleware Layer
This Layer realizes the PAAS Service described in Section 2.1. Note that Cloud Systems may limit their functionality to an IAAS service without providing a software platform layer, enabling consumers to just rent computing infrastructure. Alternatively, they may not provide IAAS service at all, bringing out a distributed software platform (PAAS) for deploying applications in the Cloud infrastructure (Google), or may provide both - IAAS and PAAS as combined services (Amazon).

Software layer
This Layer realizes the SAAS Service described in Section 2.1.
SaaS is at the highest layer and features a complete application offered as a service, on demand, via multitenancy — meaning a single instance of the software runs on the provider's infrastructure and serves multiple client organizations.

3. Service Architecture in Existing Cloud Systems

3.1 IBM

IBM offers a Cloud System providing IAAS Service. Their solution suggests mainly IBM physical hardware, where virtualization is provided on hardware level by IBM's existing products, thus achieving effectively configured environment. To enable support for heterogeneous systems, IBM allows participation of the popular x86 architecture in the physical layer, where the virtualization is done on software level. Management is realized by existing IBM and freeware tools.

Physical Layer in IBM

- **System p** – RISC/UNIX based products: servers / workstations

This is a product line of supercomputers, supporting many simultaneous operating systems. Hardware virtualization is supported: DLAR (Dynamic Logical Partitioning) with Virtual I/O and Micro-partitioning [2], [5]. For a relative idea on the hardware, here is a presentation of the Power 6 processor of this product line: "*IBM introduced during the SuperComputing 2007 (SC07) conference in Reno a new POWER6 based System p 575 with 32 POWER6 cores at 4.7GHz and up to 256GB of RAM with water cooling*" [6].

- **System z -** this type of computers are the so called "*Mainframe computers*" [2], [7], [8], where "*z*" stands for "*zero downtime*" – the systems are built with spare components capable for hot failovers to ensure continuous operations [8].

- **System i** – IBM minicomputers, supporting LPAR (Logical Partitioning) [2], [9]
- **x86** – not-IBM vendor's architecture, but it can participate in the physical layer, to support heterogeneous system requirements of the applications [2].

Virtualization in IBM

- **Systems z** is virtualized with **Processor Resource Systems Manager (PR/SM)**, which is a hardware partitioning capability. Over PR/SM could be additionally installed **z/VM**, which is software-based virtualization [2], [3].
- **System p** and **System i** are virtualized with **Power Hypervisor** [2], [7].
- **x86** is virtualized with **VMware** (which is software based virtualization). As additional information on virtualizations over the x86 architecture: "*The x86 processor architecture did not originally meet the Popek and Goldberg virtualization requirements. As a result, it was very difficult to implement a general virtual machine on an x86 processor. In 2005 and 2006, extensions to their respective x86 architectures by Intel and AMD resolved this and other virtualization difficulties.*" [4].

Infrastructure Management in IBM

- **Automatic provisioning** – for this purpose IBM uses own existing products like **Network Installation Manager** and **IBM Tivoli Provisioning Manager**, which supports writing of subsequent steps (workflows) for automatic installation and configuration of new servers [2].
- **Monitoring, Capacity planning** – this task is handled by again by IBM existing products like **IBM Tivoli Monitoring**, **Performance and Capacity Estimation Service**, **IBM Enterprise Workload Manager** [2]. For **x86** platforms virtualized with **VMware**, the management is handled with **Xen** [10].
- **Client interface** is provided via Web portal, where the user can manually submit quires for reservation and provisioning of hardware resources [2].

3.2 Amazon

Amazon has built their own public Cloud System that is officially running and offers concrete services. For Amazon, there is no public information about the system implementation, so in our study we will limit the descriptions to the extent of services provided to the consumers. Still, such an overview would provide idea and understanding about the boundaries of Cloud System and the value of its ideology.

The services provided by Amazon, could be classified as IAAS services (i.e. consumers can rent computing infrastructure), and PAAS services – environmental services (platform) for the of client applications deployed in the Amazon Cloud [11].

Elastic Cloud Computing (EC2)
IAAS Service. This is a virtual computing environment, enabling the customers via a web interface to "rent" servers, to choose operating systems to be installed on them, to manage the access rights. The response time from system side to such queries is quick enough (about few minutes) to consider this as frequent user operation, so that service users can skip the precise capacity planning, since they have the flexibility to quickly reserve and release machines if the load increases or goes down. For additional convenient, ready server configurations could be saved by the users in the form of the so called AMI (Amazon Machine Image), and installed multiple times on new machines. In addition, Amazon claims to assure high stability and availability via internal realizations of failovers and replications.

Simple Storage Service (S3)
PAAS Service. S3 could be considered as a huge hard drive somewhere in the "cloud". It provides the ability for storing and retrieving of unstructured data of size from 1 byte to 5 gigabytes. In the terms of S3, information is saved as an *object* (which could respond to a *file* in the terms of the standard file systems), and objects could be organized in the so called *buckets* (which acts similarly to the directories). A user can have up to 100 buckets and unlimited number of objects in each bucket. An object could be considered as three-parts element – key, value and metadata. The key is the name of the object (or the file name), the value is the content of the file, and the metadata is represented by key-value pairs with arbitrary content describing the object.

The data is accessed from anywhere in the "cloud" or from any place with access to it.

In summary, the service represents a "flat" file system for simple storing of persistent unstructured data (of course every programmer could organize in a programming way a hierarchical directory structure if really necessary), but in general, for the purpose of more complicated structuring of data Amazon dedicates the next service (Simple DB) providing database functionality.

Amazon Simple Database Service (SimpleDB)
PAAS Service. As name tells, this is a simplified basic database functionality, motivated by the convenience and ease of usage in counterbalance to the complicated interfaces and fine-tunings required for working with a fully functional rational database, where the user (developer) must thoroughly design the data structures, explicitly organize the indexing and take care to perform settings for efficiency. Although the usage is simplified, the functionality still provides fundamental database methods – inserting, fetching, deleting, updating and searching on basic logical filters, which in most cases covers all requirements for most of the applications. Amazon pronounces that when a fully functional database is needed, MySQL database could be freely installed on the EC2 rented machines.

Again the horizontal scaling and the reliability remain hidden inside the "cloud", and the user does not care about this part of securing its data.

Amazon Simple Queue Service (SQS)
PAAS Service. This service provides reliable, scalable queue of messages that can be used by developers to exchange information in distributed parts of their applications for execution of separate tasks. The messages are simple text messages, up to 8kb of size. They are kept up to four days or until being read and processed. When someone reads a message, it gets locked by the system to not allow someone else to read it and duplicate the task execution related its processing. If the message processing fails, it reappears again in the queue, thus if the processing component has crashed, another one in the distributed system would be able to read it and fulfill the job.

3.3 Sun

Sun's strategy is not to build and deploy a concrete Cloud System offering a concrete service, but to be a solution provider executor of building public and private Cloud Systems ordered by other companies, considering and adjusting the realization with the specific requirements. Thus the combination of components realizing each architecture level may vary [12].

Physical Layer in Sun
SPARC (from Scalable Processor Architecture) – RISC processors by Sun [12], [13]

- **SPARC** (from Scalable Processor Architecture) – RISC processors by Sun
- Other kind of hardware should be virtualized on OS level [12].

Virtualization in Sun

- **Sun xVM Server** – this is a hardware Hypervisor of Sun, Xen-based [?], that uses Solaris as a core of operating system. xVM Server includes as well the needed functionality for monitoring and management of the simultaneous guest operating systems (see below) [12], [14].
- **OS (Solaris Containers)** – Core of an operating system (similar to a limited Linux), that provides access for network virtualization on OS level, as well as allows installing of isolated "guest" operating systems – Linux, windows, Solaris on a single physical layer [12], [15].
- **Network (Crossbow)** – software implementing virtual network adapters over the physical one, virtual switches, excluding IP zones [12], [15].

Management Layer in Sun
The famous **Sun Grid Engine** is used for reserving resources for specific amount of time [12].

3.4 Google App Engine

Google App Engine Overview
PAAS Service. Google App Engine provides the ability web applications to be deployed in the Google infrastructure. Based on a typical Cloud technology, the engine uses large set of underlying servers to install, execute, multiply (when scaling is needed) the applications, to write their persistent data in distributed storage. The strategy of Google is to perform all distributing operations automatically – measurement, scaling, load balancing, hiding everything related to the complicated support of an application in a distributed environment from the user (developer). A developer can simply “throw” his application in the “cloud” and then can access its functionality assured it is reliably running without caring what really happens inside the “cloud”.

Software equipment and support for Google App Engine applications
Supported programming languages in the Google environment are **Python** and recently support was added for **Java** [16], [17].

For applications developed in both languages are valid common restrictions like:

- forbidden writing to the file system. The applications instead, could use a programming interface to the service “Datastore” (see further)
- the code executed can only be inside a HTTP request, i.e. they can not fork threads. In addition, the thread processing each request must finish in maximum thirty seconds, otherwise gets killed by the system. This is motivated by preventing the system from getting overloaded by endless cycles as well as to encourage the developers take care about the performance of their applications.

For both languages there is a respective SDK (Software Development Kit), which could be installed locally by the users for developing their web applications. The SDK simulates the Cloud environment where the applications will actually live, so that it could be convenient for the developers to write and test their applications having the real restrictions and additional services of the Cloud.

Once an application is ready, the SDK provides means for transferring it and installing into the Cloud. When a new version of the application is developed, it is again transferred to the Cloud, but does not replace the old one automatically. Instead, the user can perform his final tests in the real-life environment, and having additional administrative web interface, to review his list of applications and administer which version to be publicly active.

Additional programming tools in the Google App Engine platform
The Google Application Engine provides a set of services relative to a web environment and coordinated with standard Google-innovated services, that can be used by hosted applications:

Datastore

This service represents a database with own GQL syntax, which is a SQL-like language, but applies some basic limitations. The service is based on the existing Google elaborations for distributed data – Big Table and GFS [18]. The limitations of GQL are [17]:

- select statements could be executed on a single table only, i.e. table joins are not permitted. This facilitates and improves the performance in a distributed storage, and still should be enough for most of the applications use cases.
- simple where clauses – limitations for the >, >=, <, <= logical operations to be allowed over a single column only. This is related to the indexing efficiency, and the Google's main purpose is to provide everything as simplified and efficient as possible.
- restriction to maximum 1000 read rows per query

Google Accounts. This is a tool for organizing the users of a web application – creating and management of users, granting administration rights, etc.

URL Fetch. This is the famous Google-Search service, provided for usage by the web applications, which will be served by the same search engine providing a high speed internet search.

Mail. Service for sending email messages provided for usage by the web applications. Based on the existing GMail service.

Memcache. This service provides a ready-to-use implementation of memory cache with key-value organization. Its greatest advantage is the ability to be accessed by multiple instances of the application.

Image manipulation. Implementation of methods for images processing in JPEG and PNG format – resize, crop, rotate, etc.

Scheduled Tasks. A tool for scheduling tasks to run at regular intervals.

3.5 Microsoft

PAAS Service
Although Microsoft's suggestion could be classified as PAAS just like Google's, the functionality extent is totally different. While Google apply total restrictions, hiding all common tasks into the cloud, Microsoft offers far more extended opportunities in its application environment as well as provides to the user control over the distributed infrastructure and scale configuration of the applications.

At the time of this publication the Microsoft platform is still not ready and rolled out, but there are existing SDK and Development Tools for application elaboration [19].

Realization of the Cloud Layers in Microsoft
The platform is consisted **physically** of computers running **Windows Server 2008** operating system, on the top of which **virtualization** is implemented with **Hyper-V** isolating the environments of the separate applications [20]. The **Infrastructure Management** of the platform is proudly called **Cloud Operating System** and is entitled **Windows Azure** [21]. The core serves (according to the terms introduced by Microsoft) as a Fabric Controller for the whole datacenter of physical computers and virtual logics on their top. It monitors the status of applications, restarts them on application crash or migrates them to another machine; monitors the status of the machines as well, restarts them on failures or marks them for human check and repairing; decides where it is a suitable free place to install an application of new user.

Just like the operating system of a Personal Computer which allocates and manages the resources of the PC hardware, Windows Azure as an operating system of the Cloud, allocates and manages the resources in the large "farm" of computers. The file system is distributed and could be considered as a large bus accessed from all instances of the application. Data is triple replicated, and if a node containing one of the copies crashes, the data is copied to another node to keep the triple replication and respectively the reliability of data.

Microsoft platform characteristics and functionality extent
The platform allows the user to explicitly specify the hardware requirements for its applications and to decide how many instances of its application to have active. This somehow goes beyond the PAAS services and combines them with IAAS features. Users have the ability to decide when the applications need to scale up or down and dynamically to change the number of active instances, which in turn reflects to the their expenses on the used service, thus the realization strictly follows the Cloud slogan "*pay as you go*". To make a distinction, the Google's design hides all opportunities from the user for explicit fine tuning of the application scaling, instead - everything is performed automatically. Google's approach makes the service much easier in one hand, but cuts features on the other. This brings to a number of negative sides – the platform cannot host relatively sophisticated applications, it is hard to migrate an existing application to such a restricted environment – in most cases redevelopment would be required, and it is unclear and disputable if the automatic scaling is correct and desired to happen transparently. Some may consider this as a security hole, since an intentional force attack to the application would impact in automatic uncontrollable multiplying of the application in the Cloud, which would affect to the expenses for the occupied infrastructure. In the same time, it is more than convenient for an application if the host platform could undertake most of the common tasks. So both platforms will have their set of consumer developers, regarding the requirements they wish to fulfill.

In the platform are provided the so called Azure Services, available via a programming interface (API) over standard interprocess communication protocols like REST, HTTP, XML [20].

Live Services. A set of various standard services for processing of user information – contacts, pictures, administrative center, text redactors, e-tables, etc.

SQL Services. Analogically to the Amazon's S3 and Google's Datastore services, Microsoft also provides database service, but for distinction - this is support for a fully functional rational database. Behind the service stands the well known Microsoft SQL Server, but inside the Cloud is provided only part of the original functionality.

.NET Services. Users can rely on the well known .NET framework for deploying applications written in C# and Visual Basic.

SharePoint Services. Tools for creating web pages – blogs, wikis, etc.

Dynamic CRM Services. Software package for Customer relationship management.

Debugger. The SDK provided with the platform, contains local debugger. In the real Cloud, there wont be means for real-life debugging, instead the platform provides a solid logging system and the good logging is encouraged in its "Best Practices" for localizing the problems appearing in the real cloud environment.

4. Conclusions and future work

In this study we have elaborated on Cloud Systems concept, provided a general definition and general architecture of a Cloud System, analyzed all architecture levels and explored concrete realization examples to get a clear picture of what a Cloud System is.
Considering we explore a new technology, it is reasonable to compare it to similar well-known systems, to analyze the functional extensions of something already familiar and realize the natural growth and evolution of computing technologies. It can not be hidden that Cloud Systems inescapably remind of the familiar Grid Systems [22], so let us elaborate more on this in our conclusion chapter.
Cloud Systems are often compared to the well-known Grid Systems because of the similarity in the underlying physical resource base - farms of computers joined in a network and logically organized in a common system. Nevertheless, both types of systems are targeting completely different benefits. Grid Computing evolves in scientific supercomputing, whereas Cloud Computing is about non-supercomputing applications. I.e. Grids mainly support the execution of end-users applications as computational activities, while Clouds are mainly used for the remote deployment of services. Regarding the functional benefits they target, they do not compete. But do differences based on the functionality or usage, impose technology level differences? Undoubtedly, the use-cases do drive significant distance in technology and implementation. Grid Systems tend to have sophisticated queuing and job prioritization mechanisms, while Cloud Systems deny accepting of users if resource capacity does not allow serving the users without delay. From Cloud's point of view, this simplifies the implementation of batch mechanisms for processing queued jobs, but imposes additional monitoring and capacity planning capabilities as

internal functionality requirements. Clouds are mainly offering a real-time allocation mechanism as opposed to the best effort from Grid System. Cloud Systems concentrate on on-demand scaling up and down for service allocation and pay-as-you-go as opposed to the job execution on shared resources. The company offering Cloud System needs to over-provision based on some prediction model. If a Cloud does not scale up on user demand and as written in the Software License Agreement, it fails. In Grid, there is more transparency on resource availability and the average number of resources is typically known. If Grid users wait in the queue, they do not complain given the fact they get what the Virtual Organization agreed for.
Here we reach to the other difference that emerges from the functional use cases, and this is the User Management. Grid Systems users are usually sets of users organized in Virtual Organizations (credentials are typically enriched with VO-related information), while Clouds support individual users and introduce totally different paradigm for access rights functionality - user accounts are assigned with resources allocated for the particular user that occupies and releases them on demand.
Regarding the different resource sharing concepts, another implementation distance between both systems appears to be the Virtualization Layer. Although Grids are moving towards in using virtualized resources, virtualization is beneficial but not strictly needed. The way in which resources are dynamically occupied and reserved in the Cloud, makes the virtualization technology vital.
Focusing back to the Cloud Systems, an interesting direction for future research seems to be the investigation of the PAAS service level. A reasonable question could be targeted for many ‘standalone’ software platforms – like Tomcat [23], OSGI platforms [24], etc. - how could their functionality be implemented as a Service in the Cloud concept? For instance a big Tomcat Cloud where millions of users can “throw” their servlets and access the functionality from any place beholding the Cloud.

References

1. Wikipedia : Cloud Computing, http://en.wikipedia.org/wiki/Cloud_computing
2. IBM : Seeding the Clouds: Key Infrastructure Elements for Cloud Computing, ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/oiw03022usen/OIW03022USEN.PDF
3. IBM : System z9 Processor Resource/Systems Manager Planning Guide, http://www-01.ibm.com/support/docview.wss?uid=isg22526a41d800842b98525701c007281db&aid=1
4. Wikipedia : Virtual machine, http://en.wikipedia.org/wiki/Virtual_machine
5. IBM : PowerVM Virtualization on IBM System p: Introduction and Configuration, http://www.redbooks.ibm.com/redbooks/pdfs/sg247940.pdf
6. Wikipedia : IBM System p, http://en.wikipedia.org/wiki/IBM_System_p
7. IBM: Introduction to the New Mainframe: z/OS Basics, http://www.redbooks.ibm.com/redbooks/pdfs/sg246366.pdf
8. Wikipedia: Mainframe Computer, http://en.wikipedia.org/wiki/Mainframe_computer
9. Wikipedia: IBM System i, http://en.wikipedia.org/wiki/System_i
10. Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauery, Ian Pratt, Andrew Warfield: Xen and the art of virtualization : ACM Symposium on Operating Systems Principles, ACM, New York 2003, http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf
11. Amazon Web Services, http://aws.amazon.com

12. Sun : A Guide to Getting Started with Cloud Computing, http://www.sun.com/offers/docs/cloud_computing_primer.pdf
13. Wikipedia: SPARC, http://en.wikipedia.org/wiki/SPARC
14. Wikipedia: Sun xVM, http://en.wikipedia.org/wiki/Sun_xVM
15. Wikipedia: OpenSolaris Network Virtualization and Resource Control, http://en.wikipedia.org/wiki/Sun_crossbow
16. Google : Google App Engine, http://code.google.com/appengine/docs/whatisgoogleapp engine.html
17. Wikipedia: Google App Engine, http://en.wikipedia.org/wiki/Google_App_Engine
18. Todd Hoff: Google Architecture, http://highscalability.com/google-architecture
19. Microsoft: Windows Azure Software Development Kit, http://www.microsoft.com/downloads/details.aspx?FamilyID=b44c10e8-425c-417f-af10-3d2839a5a362&displaylang=en
20. Wikipedia: Azure Service Platform, http://en.wikipedia.org/wiki/Windows_Azure
21. Microsoft: Azure Services, http://www.microsoft.com/azure/services.mspx
22. Wikipedia: http://en.wikipedia.org/wiki/Grid_computing
23. Apache: Apache Tomcat, http://tomcat.apache.org/
24. OSGi Aliance, http://www.osgi.org

Grid Computing and Databases

Iosif Schwertner, Krasimira Schwertner
Sofia University, Department of Economics

1 Introduction

In simplest terms, grid computing is the pooling of all IT resources into a single set of shared services for all enterprise computing needs. Grid computing infrastructure continually analyzes demand for resources and adjusts supply accordingly.

The organization has not to worry about where their data resides, or what computer processes their request. It requests information or computing power and have it delivered—as much as it want, whenever it want.

This is analogous to the way electric utilities work, in that you don't know where the generator is, or how the electric grid is wired. You just ask for electricity and you get it.

Grid computing is the on-demand sharing of computing resources with in a tightly-coupled network. For those who are old enough to remember data processing in the 1980s, the IBM mainframes are a primitive example of Grid computing. Mainframes have several CPUs, each independent, and the MVS/ESA operating system allocated work to the processors based on least-recently-used algorithms and customized task dispatching priorities. Of course, RAM and disk resources were available to all programs executing on the huge, monolithic server.

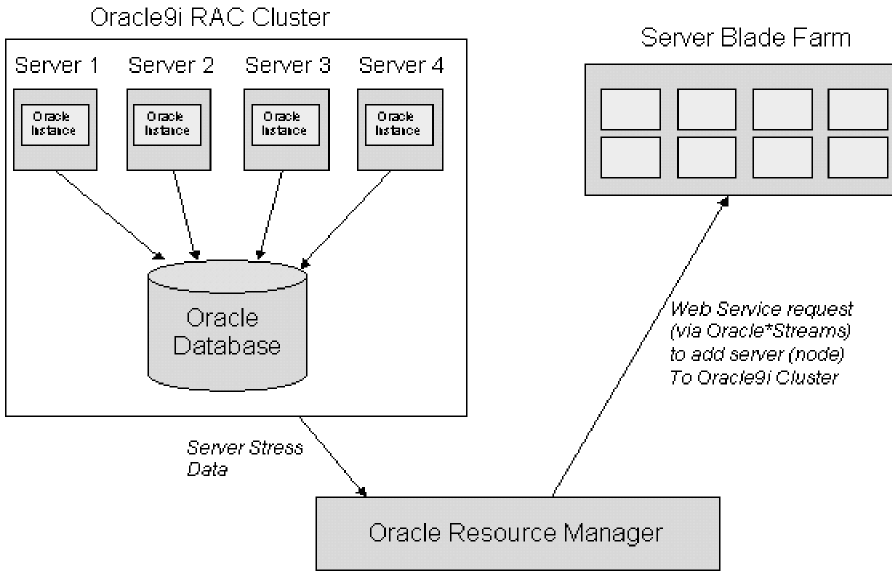
However, Grid computing is fundamentally different from mainframes. In the 21st century, Grid computing performs a "virtualization" of distributed computing resources and allows for the automated allocating of resources as system demand changes. Each server is independent, yet ready to participate in a variety of processing requests from many types of applications.

Grid computing also employs special software infrastructure (for Oracle, using Oracle Streams) to monitor resource usage and allocate requests to the most appropriate resource. This enables a distributed enterprise to function as if it were a single supercomputer (refer to Figure 1). Remember, Grid computing is all about reducing costs and ensuring acceptable performance, and infinite scalability.

2 The History of Grid Computing

The idea of Grid computing arose from the need to solve highly-parallel computational problems that were beyond the processing capability of any single computer. These types of scientific parallel computing problems are non-linear, and therefore ideal for splitting-out into subprograms because there is no need for each subprogram to communicate with the master program. The main program will send out a program to a remote computer in the network, where the program will execute independently from the master. Upon completion, the subprograms deliver the results back to the main program.

The Search For Extraterrestrial Intelligence (SETI) project is the best-known example of distributed grid computing. Participants register their PC with SETI, and the SETI software uses computing cycles from thousands of PCs across the globe to solve complex analytical problems.

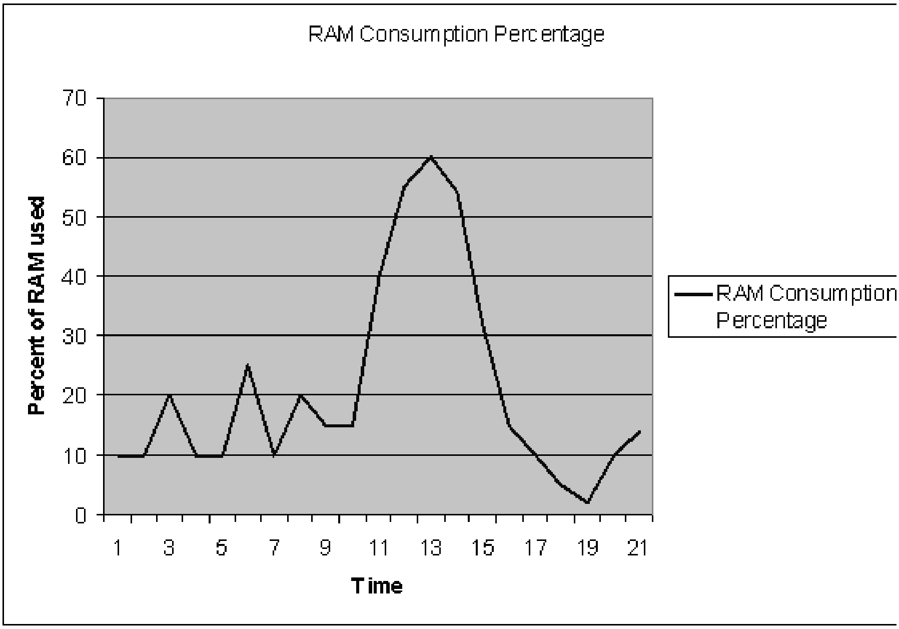


3 Why Grid Computing?

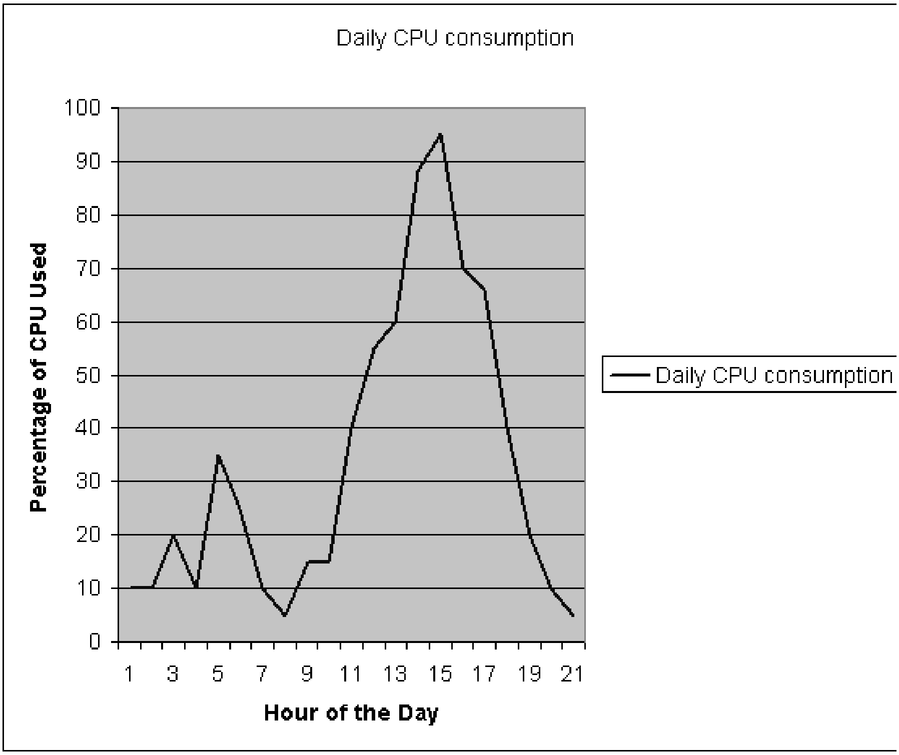
Let's take a quick look at some of the problems that led to the development of Grid technology. The trend of the 1990s was toward small, cheap, dedicated servers, and Oracle shops replace a single mainframe with hundreds of UNIX and MS-Windows servers. However, IT shops soon realized that the high expense of maintaining multiple servers plus the requirement to over-allocate hardware was eroding these savings. The problems of using dedicated servers include:

1. **Expense**: In large enterprise data centers, hardware resources are deliberately over-allocated to accommodate processing-load peaks.
2. **Wastefulness**: Because each application resides on a single server, there is significant duplication of work, and a sub-optimal utilization of RAM and CPU resources.
3. **High Maintenance**: In many large shops, a "shuffle" occurs when an application outgrows its server. A new server is purchased, and the database is moved to the new server. Then the old server becomes available, and another database is migrated onto the old server. This shuffling of databases between servers is a huge headache for the administrators who are kept busy after hours moving databases to new server platforms.

In Figure 2, we see how each dedicated Oracle server must be over-allocated for RAM to meet system demand. As we know, an increase in volume requires additional PGA RAM for each connected Oracle user, and the savvy DBA will allocate RAM to the high-water mark of RAM usage.



We see the same type of over-allocation for CPU resources. To ensure acceptable response time at peak usage, the Oracle DBA will assign additional CPU resources to each server (refer to Figure 3).



4 How Grid Works with Oracle?

Oracle Grid computing allows for two ways to provide on-demand computing resources. These involve direct hardware allocation via blades, and the allocation of remote servers to execute Oracle tasks:

- **Add processing power to existing systems** — Server blades can be allocated to Oracle9i RAC clusters.
- **Server banks** — Oracle servers can be established with no local data and copies of all application stored procedures. This would allow for three types of distributed execution:
- **Remote execution** — Entire programs, along with the data are shipped to the remote server for execution. Using Oracle Streams, the results are sent back to the initiating instance.
- **Remote data extracts** — Distributed SQL can execute from a bank of servers with instances, but no local Oracle data
- **Remote process execution** — Oracle applications can call PL/SQL stored procedures on the least-loaded servers

Oracle follows the Sun Microsystems definition of delivering computing resources as a "utility," much the same way as additional electricity is added to a network during high-usage times.

Oracle whitepapers suggest that the next generation of Oracle Grid computing will use Oracle*Streams as the glue for the Grid communications. Oracle*Streams allows for the streaming of data between Oracle instances, and also provides built-in tools for replication, data warehouse ETL (data loading), message queues and messaging.

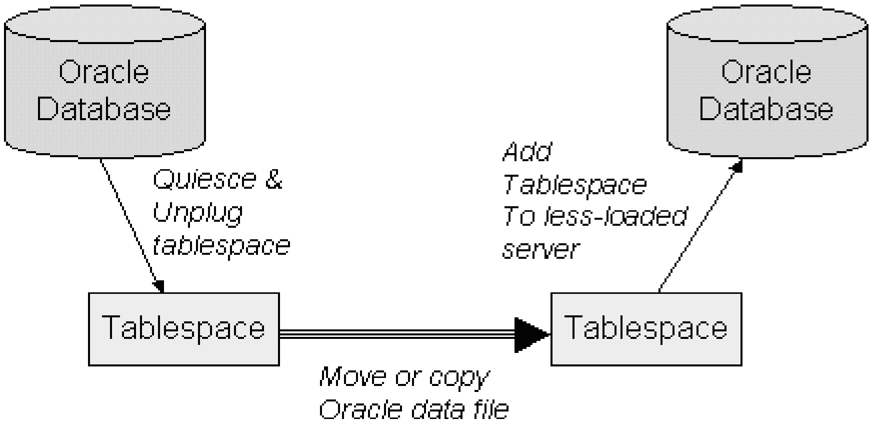
Hence, Oracle*Streams could be used as the vehicle for a Grid system whereby Oracle detects a resource shortage and invokes a procedure to relieve the stress on the database. While using Oracle*streams as the backbone, Oracle identifies three ways that computing resources can be assigned:

6. Oracle Real Application Clusters (RAC)

Using existing Oracle RAC capabilities, data blade servers can be used to add instance nodes. This allows for automated scalability, and a rack of server blades can be allocated to the Oracle9i RAC system on an as-needed basis.

7. Oracle transportable tablespaces

Transportable Tablespaces allow Oracle data files to be unplugged from a database and copied to another Oracle instance (on another server) and then added into that Oracle instance (refer to Figure 5). Transportable Tablespaces also supports simultaneous mounting of read-only tablespaces by two or more databases. Using transportable tablespaces with Oracle*Streams, the Grid could request a refresh of the tablespace data at any time.


Oracle Database
Quiesce & Unplug tablespace
Add Tablespace To less-loaded server
Oracle Database
Tablespace
Tablespace
Move or copy Oracle data file

8. Distributed SQL

Oracle SQL can be executed from any Oracle instance, **accessing the data remotely** using **database links**. This provides a framework where SQL servers may be allocated, all executing queries that access the same Oracle database.

However, it is important to remember that not all types of queries will benefit from remote execution. A simple Oracle query such as the one below would have all of the processing work (sorting) done on the remote server, saving no computing resources.

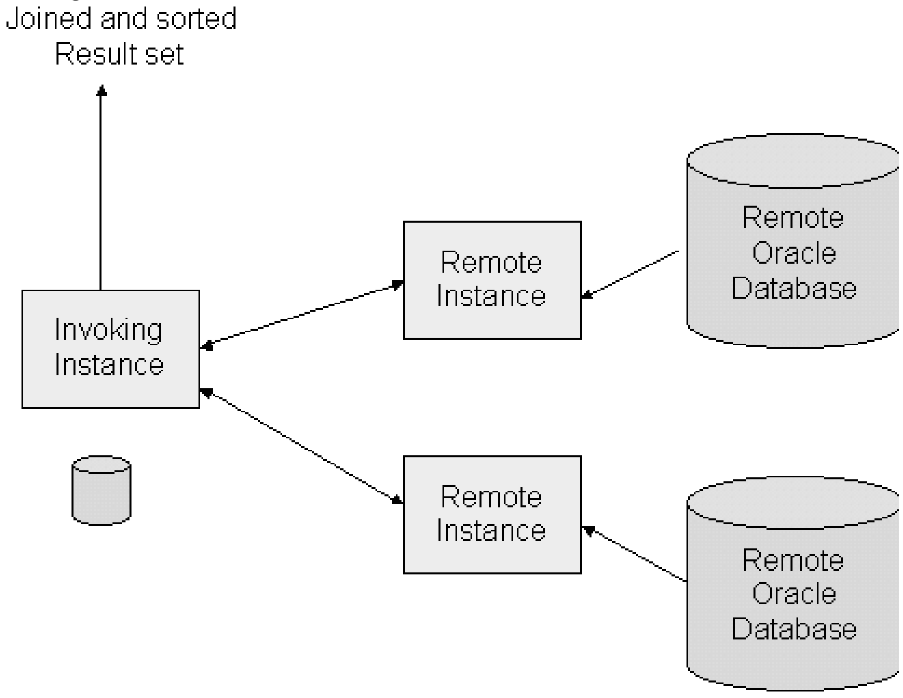
```
select
   cust_name
from
```

```
   customer
order by
   cust_name;
```

Farming out SQL queries to remote non-RAC instances makes more sense when the initiating instance has an opportunity to use its own computing resources to service the query. For example, consider the following distributed query, run from an instance named Omaha:

```
select
   cust_name.
   order_number,
from
   customer@new_york,
   orders@san_fran
where
   region = 'WEST'
order by
   cust_name,
   order_number;
```

This type of query is ideal for remote execution because the bulk of the data processing will be done on the Omaha instance.



References

1. Oracle® Real Application Clusters Administration and Deployment Guide 11g Release 1 (11.1), Part Number B28254-05, Oracle Corporation
2. Oracle Grid Computing, An Oracle Business White Paper, October 2008