Journal of Applied Computer Science Methods

Published by University of Social Sciences

ACSM

Volume 5 Number 1 2013

University of Social Sciences, IT Institute

ISSN 1689-9636

International Journal of Applied Computer Science Methods

Journal of Applied Computer Science Methods

Published by University of Social Sciences

ACSM

Volume 5 Number 1 2013

University of Social Sciences, IT Institute

INTERNATIONAL JOURNAL OF APPLIED COMPUTER SCIENCE METHODS (JACSM)
is a semi-annual periodical published by the University of Social Sciences (SAN)
in Lodz, Poland.

PUBLISHING AND EDITORIAL OFFICE:
University of Social Sciences (SAN)
Information Technology Institute (ITI)
Sienkiewicza 9
90-113 Lodz
Tel.: +48 42 6646654
Fax.: +48 42 6366251
E-mail: acsm@swspiz.pl
URL: http://acsm.swspiz.pl

AIMS AND SCOPE:

The **International Journal of Applied Computer Science Methods is** a semi-annual, refereed periodical, publishes articles describing recent contributions in theory, practice and applications of computer science. The broad scope of the journal includes, but is not limited to, the following subject areas:
Knowledge Engineering and Information Management: *Knowledge Processing, Knowledge Representation, Data Mining, Machine Learning, Knowledge-based Systems, Knowledge Elicitation, Knowledge Acquisition, E-learning, Web-intelligence, Collective Intelligence, Language Processing, Approximate Reasoning, Information Archive and Processing, Distributed Information Systems.*
Intelligent Systems: *Intelligent Database Systems, Expert Systems, Decision Support Systems, Intelligent Agent Systems, Artificial Neural Networks, Fuzzy Sets and Systems, Evolutionary Methods and Systems, Hybrid Intelligent Systems, Cognitive Systems, Intelligent Systems and Internet, Complex Adaptive Systems.*
Image Understanding and Processing: *Computer Vision, Image Processing, Computer Graphics, Pattern Recognition, Virtual Reality, Multimedia Systems.*
Computer Modeling, Simulation and Soft Computing: *Applied Computer Modeling and Simulation, Intelligent Computing and Applications, Soft Computing Methods, Intelligent Data Analysis, Parallel Computing, Engineering Algorithms.*
Applied Computer Methods and Computer Technology: *Programming Technology, Database Systems, Computer Networks Technology, Human-computer Interface, Computer Hardware Engineering, Internet Technology, Biocybernetics.*

DISTRIBUTION:

Apart from the standard way of distribution (in the conventional paper format), on-line dissemination of the JACSM is possible for interested readers.

CONTENTS

ADAPTIVE FILTER FOR INERTIAL SYSTEMS

Marek Orzyłowski

IT Institute, University of Social Sciences,
9 Sienkiewicza St., 90-113 Łódź, Poland
marek.orzylowski@gmail.com

Abstract

The stochastic model of the disturbances handled by Kalman filters and the necessity of accurate identification of the dynamic model of the controlled system or process bring about a significant limitation of use of Kalman filters in practice. The paper presents filter designed for inertial systems whose models and control signals are not well known or are beyond description. The assumptions leading to a significant simplification of Kalman algorithm are described. On this basis, the algorithm with an experimentally matched parameter for the filter properties modification is introduced. An example of effective adaptive filtration is presented also.

Key words: digotal signal processing, adaptive filtration, Kalman filter, inertial systems, thermal systems

1 Introduction

Filtration is an important tool in digital signal processing. The adaptive filtration of signals describing the state of the systems and processes in the presence of noise and measurement inaccuracies has been used for a long time. Many algorithms are employed for this purpose (e.g. [1]), including the Kalman filter algorithm. Unfortunately, the stochastic model of the disturbances handled by Kalman filters and the necessity of accurate identification of dynamic model of the controlled system or process bring about a significant limitation of use of Kalman filters in practice. The paper presents an attempt to get around these limitations by using a modified Kalman filter algorithm for adaptive filtration.

This filter is designed for inertial systems whose models and control signals are not well known or are beyond description. An example of such a system can be a thermal system with accidental batch. Another example is the non-explosive combustion of a sample of unknown composition. The filter described in the article has been, among others, applied for the measurement of the SO2 and CO2 contents in combustion gases produced in such a process [2].

2. Kalman filter

The adaptive properties of Kalman filter [1] are associated with the optimal linear quadratic estimation of the system state, based on the knowledge of the system model, on the control vector and on the parameters of stochastic disturbances of state and also on the measurement of state variables. The algorithm is based on a set of equations describing the dynamics in the state space in the presence of disturbances. Its discrete form is

$$\begin{aligned} x(k) &= Ax(k-1) + B(k)u(k) + v(k) \\ y(k) &= Cx(k) + w(k) \end{aligned} \tag{1}$$

where A – state matrix, B – input matrix, C – output matrix, x – state vector, y – output vector, u – control vector, v – state disturbances vector with the covariance matrix Q, and w – measurement disturbance vector with the covariance matrix R. For a better legibility we shall assume later on that the state vector is fully observed, which means that C is a unitary matrix. It does not cause any limitation, because on the basis of the full form of the Kalman filter description the necessary modifications can be easily made.

The Kalman filter algorithm of the estimation of state vector value in the k-th step, made before the measurement y(k), is based on its estimation in the previous step $\hat{x}(k-1 \mid k-1)$, as described by the equation

$$\hat{x}(k \mid k-1) = A\hat{x}(k-1 \mid k-1) + Bu(k-1) \tag{2}$$

The estimation error is defined as

$$\tilde{x}(k-1 \mid k) = x(k) - \hat{x}(k \mid k-1) \tag{3}$$

It has the covariance

$$P(k \mid k-1) = AP(k-1 \mid k-1)A^T + Q(k-1) \tag{4}$$

When the measurement y(k) is completed, the additional information is obtained, on the basis of which the state vector estimator in the k-th step gets the form

$$\hat{x}(k \mid k) = \hat{x}(k \mid k-1) + K(k)\left[y(k) - \hat{x}(k \mid k-1)\right] \tag{5}$$

where K(k) is the Kalman gain matrix, which can be calculated from the equation

$$K(k) = P(k \mid k-1)\left[P(k \mid k-1) + R(k)\right]^{-1} \tag{6}$$

The covariance of the state estimation error for k step is related to the covariance (4) as follows

$$P(k \mid k) = \left[1 - K(k)\right]P(k \mid k-1) \tag{7}$$

3. The assumptions for adaptive filter

It was assumed in general that the described adaptive filter is intended for systems of inertial type. For many inertial systems in which the adaptive filtration can be used the process can be split into three phases of the filtered signal change: rising, stabilization and dropping. A typical example for this is the temperature in thermal systems: they are heated up, kept at a constant temperature and then cooled down. The presented adaptive filter is particularly destined for such a kind of work.

Another assumption was that the filtered state vector consists of only one state variable and its step-to-step changes are small over the applied measurement repetition period τm, especially in comparison with the full range variation that is physically permissible in the system.

The next assumption for the presented filter is a limited knowledge of the system model and its input control signals, which on basis of eq. (2) results in an inaccuracy of prediction of the state vector in the consecutive step. Additionally, in such a case the control signals should be treated as a part of disturbance signal. In such a situation the disturbance signal v(k) cannot be described as stochastic with normal distribution, as it is assumed in Kalman filters. However, for the filter in question, similarly like for a usual Kalman filter, the measurement disturbances will be modelled as the stochastic signal w(k) of normal distribution and known variance R.

It was possible to make use of Kalman filter in the described case thanks to some simplification of the model of inertial system and the adoption of some analogy between the signals influencing the state of the system in both filters. In the consequence, the described filter loses the optimality defined for Kalman filter. As a result of such a modification the additional weight coefficients have been applied to the disturbances. This enables us to experimentally adjust the filter properties to requirements in individual cases.

As mentioned, the presented adaptive filter has been designed for the single variable filtration only. As a result, all matrices and vectors in Kalman filter eq. (1), (7) become scalars and all covariance matrices become variances. This much simplifies the applied algorithm.

To adapt the Kalman filter algorithm to our filtration case the interdependences that would allow the heuristic equivalence of the process signals filtered in both filters should be determined.

It was assumed in the proposed solution that the step-to-step changes Δx of the state variable are small. Basing on this it can be also assumed that when the input signal u(k) and measurement signal y(k) are unknown, the most probable value of state vector in the step k is equal to its value filtered in the step k-1. The appropriate estimator can be written as

$$\hat{x}(k \mid k-1) = \hat{x}(k-1 \mid k-1) \tag{8}$$

Substituting equation (8) for (2) is synonymous with setting the system parameters as: A=1 and B=0. It had been assumed before that C=1, so the eq. (1) can be replaced by

$$\begin{aligned} x(k) &= x(k-1) + v(k) \\ y(k) &= x(k) + w(k) \end{aligned} \tag{9}$$

It is worth to stress that value A=1 describes the system of integrating character, whose pole z=1 (in continues notation s=0). For the time increments Δt small enough (a few times measurement period τm) such a model in many cases well enough approximates the dynamics of inertial systems of the time lag T>> Δt.

To interrelate the respective equations of both filters we assume that the signal v(k) in the Kalman filter equations, now signed as vn(k), corresponds to v(k) in the real system filtered by the presented adaptive filter, upon certain conditions.

The stochastic signal vn(k) can be described by its variance , which for N samples can be calculated as

$$\sigma^2(k) = \frac{1}{N-1} \sum_{i=k-N+1}^{k} \left[v_n(i) - \bar{v} \right]^2 \tag{10}$$

where $\bar{v}$ is the expected value, which for e.g. for white noise is equal zero. As mentioned, v(k) in the real system shows some time trends of unknown rate, so its parameters cannot by characterized using eq. (10). For this it was assumed that the criterion of the signals vn(k) and v(k) equivalence is equality of averaged square values over the step-to-step increments.

During Kalman filtration, the measurements of y(k) values and the estimation of state variables values are carried out. In the result, for the determination of v(k) and v_n(k) equivalence we can accept the equality of parameter described as

$$S(k)=\frac{1}{1-N}\sum_{i=k-N+1}^{k}\left[y(i)-\hat{x}(i-1\,|\,i-1)\right]^2 \tag{11}$$

Variance Q(k) for vn(k) and u(k)=0 can be easily calculated on the basis of S(k) using the Kalman filter equations. Taking into account the assumption of equivalence of v(k) and vn(k), the value Q(k) calculated similarly on the basis of v(k) can be treated likewise in proposed filter equations. The finally accepted method of calculation for S(k) and that for the substitute value Q(k) in adaptive filter is described in the next part of the paper.

The last essential assumption for the adaptive filter under consideration is the possibility of its properties modification depending on the process needs. Among others, it concerns the compromise between the requirement of low delays during fast changes of system state and that of the effective noise signal rejection in the steady state. To achieve this goal the diversification of weights assigned to the state and measurement disturbances are allowed.

3. The adaptive filtration algorithm

At the beginning let us temporarily assume that signals v(k) and w(k0 are white noise, thus they have normal distribution. For the sake of simplified form of model (9), the equations of Kalman filter for such a system also undergo simplification. They get the form

$$\begin{aligned}
K(k)&=\frac{P(k-1\,|\,k-1)+Q(k)}{P(k-1\,|\,k-1)+Q(k)+R}\\
P(k\,|\,k)&=(1-K(k))(P(k-1\,|\,k-1)+Q(k))\\
\hat{x}(k\,|\,k)&=K(k)y(k)+(1-K(k))\hat{x}(k-1\,|\,k-1)
\end{aligned} \tag{12}$$

The values of y(k) sequence are obtained from measurements. Taking into consideration (9) they are equal to

$$y(k)=x(k-1)+v(k)+w(k) \tag{13}$$

After filtration (12), we obtain the sequence of estimated values of state $\hat{x}(k \mid k)$ whose estimation error referred to the previous step is equal

$$\tilde{x}(k-1 \mid k-1) = x(k-1) - \hat{x}(k-1 \mid k-1) \tag{14}$$

and whose variance is P(k-1|k-1)

The sequences of signal v(k) and w(k) are not correlated and have zero value expected values, due to the assumption, they are white noise. Any sequences of either of them resulting from shifting by arbitrary number of steps are not correlated either. Taking into account eq, (12) and eq, (3) the equivalence parameter S(k) described by eq. (11) can be written as

$$\begin{aligned} S(k) &= \frac{1}{N-1}\left[\sum_{i=k-N+1}^{k}\bigl(y(i) - \hat{x}(i-1 \mid i-1)\bigr)^2\right] \\ &= \frac{1}{N-1}\left[\sum_{i=k-N+1}^{k}\bigl(v(i) + w(i) + \tilde{x}(i-1 \mid i-1)\bigr)^2\right] \\ &= Q(k) + R + P(k-1 \mid k-1) + Cov(k, \tilde{x}, v, w) \end{aligned} \tag{15}$$

Covariance of error $\tilde{x}(k \mid k)$ referred to signals v(k) and w(k) signed as $Cov(k, \tilde{x}, v, w)$ is for Q<R negligibly small in the total balance of estimation errors. Variance Q(k) assuming that $Cov(k, \tilde{x}, v, w)$ is equal to zero can be determined from the equation

$$Q(k) \cong S - R - P(k \mid k) = \frac{1}{N-1}\left[\sum_{i=k-N+1}^{k}\bigl(y(i) - \hat{x}(i-1 \mid i-1)\bigr)^2\right] - R - P(k-1 \mid k-1) \tag{16}$$

However, one should be aware that relationship (16) in some cases can lead to overestimation.

During Kalman filtration (12) recursive calculations are executed, among others the calculations of error variance P(k|k).

To adapt the calculations of the actual value of Q(k) variance to this method of calculation it is convenient to substitute a modified equation (17) for (16) that corrects step by step the value of Q(k). This equation is:

$$Q(k) = (1-\alpha)Q(k-1) + \alpha\left[\bigl(y(k) - \hat{x}(k-1)\bigr)^2 - R - P(k-1 \mid k-1)\right] \tag{17}$$

where α is weighting coefficient of value e.g. 0.25. Eq. (17) has the form of a low-pass IIR filter moderating the accidental fluctuations of the calculations results.

It is worthwhile to take into account the effects of Q(k) estimation with reference to R variance. For the described assumptions the Kalman gain K applied in eq. (5) has values shown in Figure 1.

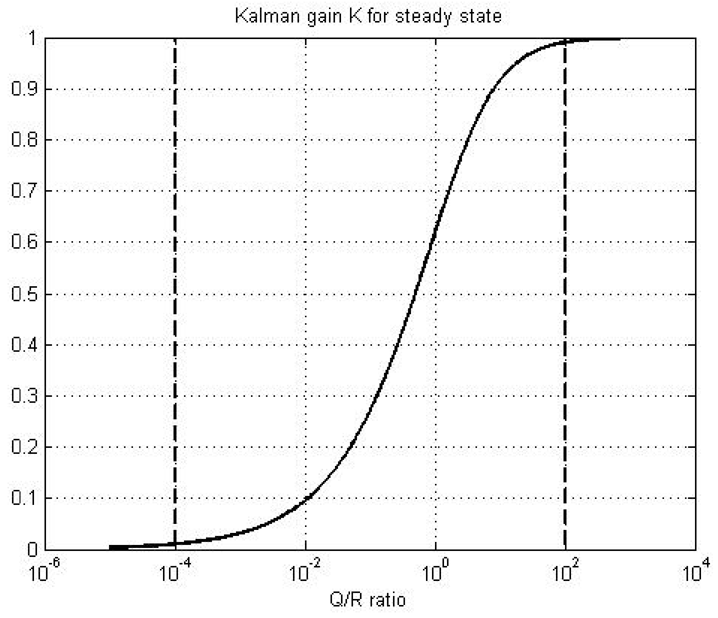


Figure 1. Kalman filter gain vs. Q/R ratio

Accidental fluctuation of Q(k), calculated from eq. (17) on the basis of disturbed measurement values, can lead to even negative values. So it is necessary to limit the lowest value of Q(k)/R to e.g. 1/10000 (K=0.01). According to Figure 1 the value of gain K is very small for this ratio. Similarly the upper limit e.g. Q(k)/R=100 (K=0.99) is also reasonable. So, we have the following limitation

$$0.0001R \leq Q(k) \leq 100R \tag{18}$$

So far, the model of the system dynamics has been considered in a simplified form (9). The state disturbances taken into consideration were treated as white noise. Additionally, the value of Q(k) derived from eq. (16) can be overestimated.

As a result, some corrections in the application of the presented equations should be made in order to obtain effective adaptive filtering in the real system. Figure 1 shows that the filter gain K depends on Q/R ratio. Let us add a new weight coefficient β in the equation describing Q(k). Then the modified value of state disturbances designated as Qm(k) can be written as

$$Q_m(k) = (1-\alpha)Q_m(k-1) + \alpha\left[\beta(y(k) - \hat{x}(k-1))^2 - R - P(k-1 \mid k-1)\right] \tag{19}$$

The weight coefficient β can reduce the effects of Q(k) overestimation and that of the poor evaluation of the measurement disturbances variance R. If β drops down, it reduces the influence of measurement disturbances on the filtered variable in the steady state, but on the other hand, leads to the time delays increase during the periods of fast changes.

The right value of β should be a trade-off based on experimental evaluation.

After the modification associated with introducing the coefficient β and substituting Qm(k) for Q(k) variance in the Kalman filter, the final equations of the presented adaptive filter take the form

$$\begin{aligned} K(k) &= \frac{P(k-1 \mid k-1) + Q_m(k)}{P(k-1 \mid k-1) + Q_m(k) + R} \\ P(k \mid k) &= (1 - K(k))(P(k-1 \mid k-1) + Q_m(k)) \\ \hat{x}(k \mid k) &= K(k)y(k) + (1 - K(k))\hat{x}(k-1 \mid k-1) \end{aligned} \tag{20}$$

The operation of the filter described by (18)-(19) is shown in Figure 1 to 3, using a test signal that was deterministic by nature, being a combination of single steps and exponents. This signal was jammed with a measuring noise of variance R. The test signal proper simulates the state changes going on at various rates and its character is much different from that of the noise with normal distribution, which Kalman filtration concerns.

Figure 2 shows simultaneously the test signal and the filtration result obtained with a LP filter having large time-constant. It allows for good filtration in the quasi-steady state, but brings in considerable delays, particularly in the phase of initial rise. Reduction of the filter time-constant much reduces the delays, but the filtration becomes practically ineffective in the quasi-steady state (Figure 3). Figure 4 demonstrates the operation of the discussed filter. We can easily notice that the adaptive filter is free of the above-described faults of the filters with an invariable time-constant. The value of β=0.5 in equation (12) was assumed for this filter.

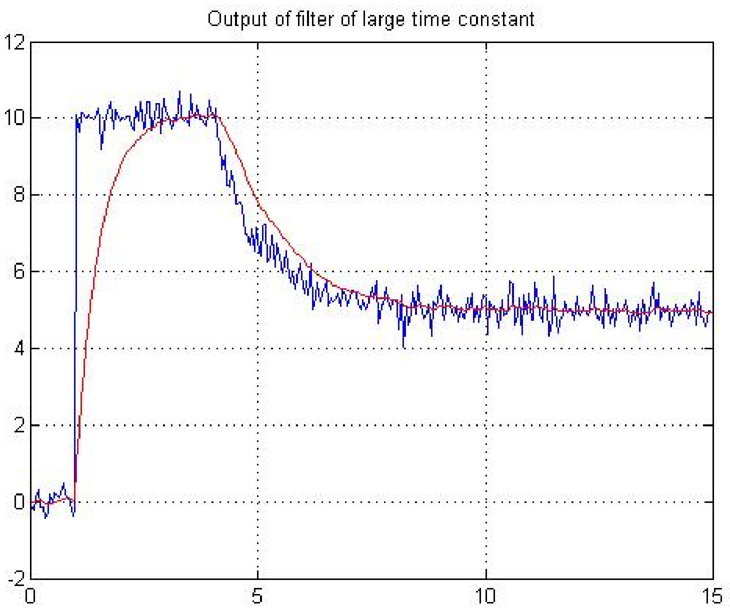


Figure 2. Filtration of LP filter of invariable large time constant

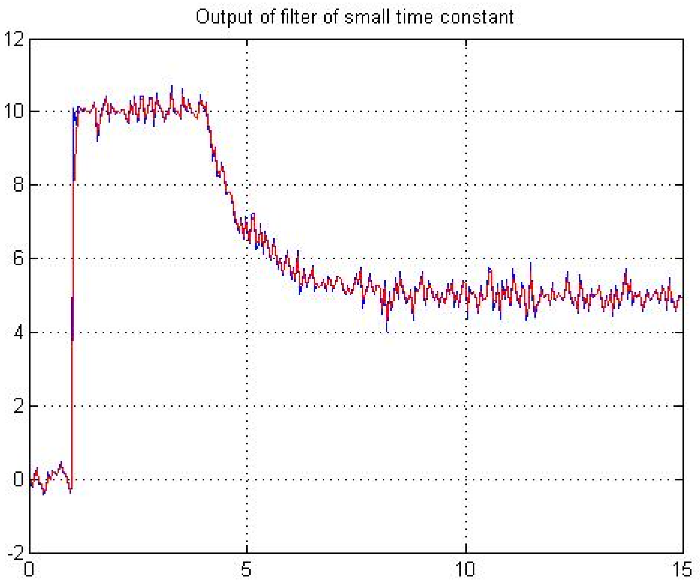


Figure 3. Filtration of LP filter of invariable small time constant

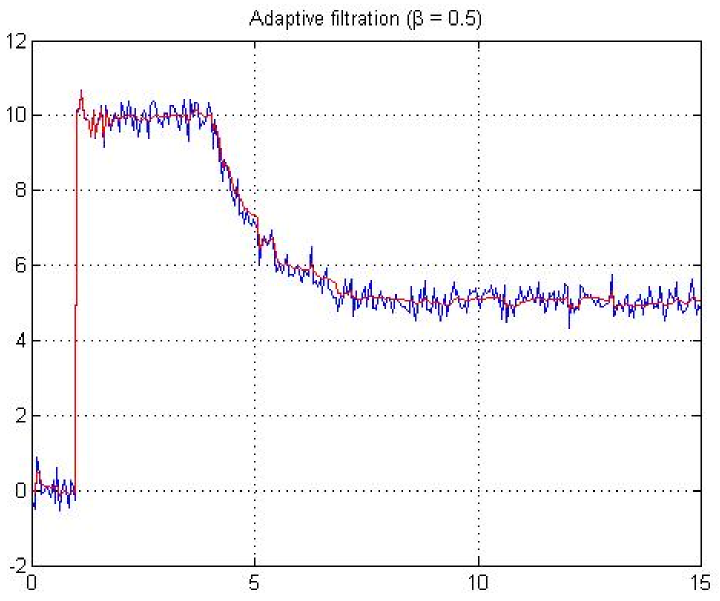


Figure 4. Filtration of adaptive LP filter

4. Conclusions

The paper presents an adaptive filter based on Kalman filter algorithm. This filter doesn't need accurate information either about the system dynamics model or about the control signal on the system input. The filter estimates the values of state variable on the basis of the measurement disturbances variance and the variations of signal on the system output. As it is shown in Figure 4 such a filter can be very effective in various phases of the real process.

The described filter was successfully applied by the author in a number of thermal systems for control (eg. [3]) and measurement purposes.

References:

1. Saeed V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*, J. Wiley, 2009
2. Orzylowski M., *Przetwarzanie sygnału pomiarowego w analizatorze zawartości siarki i węgla w popiołach,* Elektronika, Sigma-NOT, Warszawa, No. 2013/4
3. Orzyłowski M., *Process-oriented suboptimal controller for SiC bulk crystal growth system,* Elektronika, SIGMA,-NOT, No. 8/2012, pp. 11-15

THE MODEL OF THE DIGITAL TERRAIN MAP (DTM)

Andrzej Kaźmierczak

Geodesy and Cartography Institute, Social Academy of Management, Lodz, Poland
akazmierczak@spoleczna.pl

Abstract

The true digital terrain map (DTM) is needed to calculate so-called terrain gravity potential. It is one of the fundamental factors in earth geoid shape determination. There are many methods of calculation of the gravity potential. The gravity integral should cover all Earth area. There is no possibility to establish digital terrain map for whole planet. So it must be numerically proved, if the gravity integral converges to its extreme value for some limited part of the earth area. The model of DTM was established mathematically for testing a different way of gravity potential calculation. It contains all features of real DTM and has been build up in two steps processing.

Key words: numerical and functional modeling, gravity terrain potential, digital terrain map

1 Introduction

A modern information technology is widely implemented in present geodesy and cartography. The new measurements technologies like a GPS, laser and radar scanning, gravimetry and gravity gradient measurements allows the precise earth surface mapping with high accuracy that was earlier unavailable. The investigation of an earth figure is impossible by direct measurements and usually has been done by some theoretical models. The fundamental model of earth figure determination is the Stokes method and the modified Stokes – Helmert method [2,9,11,12]. There is no possibility to integrate gravity anomaly reduced to see level over an all geoid's surface, or, like in modified by Molodensky the Stokes–Helmert method, over all physical earth surface. The integration needs some approximate downward continuation of gravity anomaly to see level and surface condensation of the mass over geoid. The results are non accurate [15,16,18]. In modified method Stokes integral is made over physical earth surface gravity anomaly and the result are quasigeoid and height anomaly. The direct and indirect effect of topography must be derived to evaluate quasigeoid height over geoid. The rigorous determination of terrain mass potential is in work [8] and approximate one in work [9]. The correct determination of a topographic potential at any point on Earth needs a knowledge of the mass decomposition over all geoid surface. This is actually unrealistic, so all calculation has been done by assumption of local constant mass density and knowledge of the topographical height mass over geoid. The approximate value of a topographic potential we get by integration of point mass over all volume topographic mass by some assumption about its density. The main problem in this estimation is, how great is an approximate error. Moreover, using different Earth models (sphere, ellipsoid, plane), we may question, if simpler or more complex model of geoid is needed [13,15,16,17], to obtain acceptable accuracy. The answer rarely is possible by theoretical consideration, because of problem complexity. The solution gives a numerical calculations. The potential integration over mass needs a great base of data from area involving territories of many countries. It's accessibility is not always possible, because of the administrative law of different countries. It is difficult to estimate, how great area potential integrals must cover, theoretically over all Earth surface, to obtain sensible accuracy. The similar problems appear by attempts of improving the model of geoid, as a geodetic reference frame for calculation of the topographic potential. The calculations based on assumption of a constant terrain height are far from real situation [8]. To avoid this problems in real testing of different theoretical models, in this work there is proposed some model of digital terrain map (DTM). It may help to test some theoretical solutions of geoid determination. The similar option we find in work of Kryński J. [9].

2 The requirements a Model of the DTM

The digital terrain model must have all characteristic features of real land. It must contain a great topographical features like lowlands, highlands, plateaus, mountains, without lakes or rivers, small depth, sizes and smaller density. The theoretical requirements about size of such model are unknown, but it seems that area size 2000kmx2000km is sufficient. This mean geographical size 15o-20o from south to north and the same from west to east. Besides that great size features it is necessary to build from the ground base local small topographical structures like hills, abrupt, mountain peaks. The numerical calculation of gravity potential on Earth surface with appropriate accuracy non averaging local height must be done on grid range of 20-30m, it means 1" in geographical latitude and longitude. Only such grid will be adequate on terrain with slope inclination great than 20o (100m of distant means 20m or more of height difference). It means construction of at least $4 \cdot 10^9$ points with defined height and density. It seems reasonable to build at first complete DTM, because of time-consuming numerical arithmetic. The basic DTM is to big to put it in one memory set, so we need to divide it into smaller parts and build convenient access tools to required fragments. General coordinates of each point are two integer numbers and number of adequate part of base DTM. It's easy to transform this coordinates to geographical or geodetical longitude and latitude, and consecutively Cartesian or geocentric coordinates at any place on the Earth model (sphere, ellipsoid, plane).

3 The Representations of the Big Topographical Features

The building of topographical feature is like some kind of an artistic work and means creation of fictitious terrain images with all mentioned earlier big elements. Because of its great size first basic map is build on grid sized about 0.5kmx0.5km (16"x16"). The height on denser grid are going to be calculated algorithmically by assumption of constant inclination the planes build on rare ground grid. The plane is defined unambiguously by three points of a grid, so the base of denser grid must be a triangular net. Its going to be defined further.

Now we define ground requirements of DTM.

A look at any map of scale 1:1000000 shows presence lowlands, highlands, low (800÷1500m) and high mountains (tops over 2000m). On the large scale maps contours are relatively simple, without sharp turns. From lowlands grow up highlands, from highlands grow up mountains. The layers arrangement has multi-pyramid structure. This allows build up similar structure by mathematical methods. The Gauss function of two variables seems to be especially useful:

$$g(x,y) = h \cdot \exp(-[(x-x_0)^2 + (y-y_0)^2]/d^2) \qquad (1)$$

It doesn't has to be normalized for purpose of this work and its values shape on xy plane forms regular circular hill with center in (x_0, y_0) point and height equal h. Of course this regular shape needs some modifications, like irregular changes of slope inclination in different directions and distances from the top. The another modifications must take account, that geological structures have irregular directions against meridians and parallels. H parameter allows to control a maximal height of model's geological structure. Parameter d controls size of a structure. By assuming that 1/16 of maximal height is a border of modeled geological structure we find for this value distance 1.67d. So d parameter is good rating of structure size. We use function g defined by (1) to establish height at any DTM point.

For mathematical reason a rectangular grid was chosen as an integer coordinate base size (N+1)x(M+1). For convenience (real distance) every point coordinate were calculated in km as (x,y)=(D*i,D*j); $i\in(1,N+1)$, $j\in(1,M+1)$, D is size parameter in km (D=0.5km for fundamental map).

Next step is to randomize the regular gaussian hill. It may be achieved by some functional transformation of map area and height's evaluation $h(x,y)$ as g(x',y') at every point of DTM:

$$\begin{aligned}&[x',y']=f([x,y]);\\&[x,y]=\vec{r};\quad [x',y']=\vec{r}\,'\\&h(x,y)=g(x',y');\end{aligned}\tag{2}$$

4 The Randomized Transformation of the Map Coordinates

A first step of coordinate transformation is rotation around a center point (x_0,y_0) and elongation along one of axes. We rotate reference system by angle φ and multiple one coordinate, for example y', by $\sqrt{k}$. The result is:

$$\begin{aligned}&\begin{cases}x'=(x-x_0)\cos\varphi+(y-y_0)\sin\varphi;\\ y'=-(x-x_0)\sin\varphi+(y-y_0)\cos\varphi;\end{cases}\\&g(x',y')=h\cdot\exp(-[x'^2+k\cdot y'^2]/d^2);\end{aligned}\tag{3}$$

We obtain the last effect introducing a changeable rotation angle φ. It may be function of azimuth A, where A is angle in polar coordinates with pole (x_0,y_0). The A angle is periodic, so $\varphi(A)$ must be a periodic function: $\varphi(0)=\varphi(2\pi)$. In this work the following functions were chosen:

$$\begin{aligned}&A=\operatorname{arctg}\left(\frac{y-y_0}{x-x_0}\right);A\in<0,2\pi>\\&\varphi(A)=\varphi_0\cos A;\qquad \varphi(A)=\varphi_0\sin A;\\&\varphi(A)=\varphi_0\frac{(A-\pi)^2}{\pi^2}\cos(A)\end{aligned}\tag{4}$$

Below on Figure 1-4 are shown some effects of transformation (3) and (4) for rectangular area. We see contour graph of function (2) after transformation its rectangular domain.

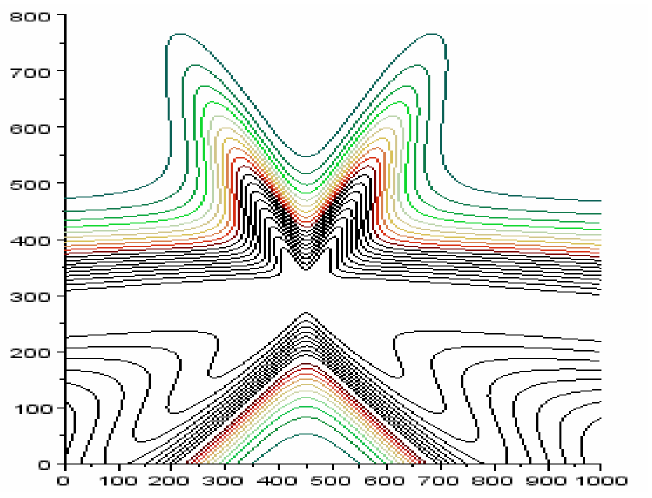


Figure 1. elongation k=4, φ0=1.5, d=200, φ=φ0 cosA

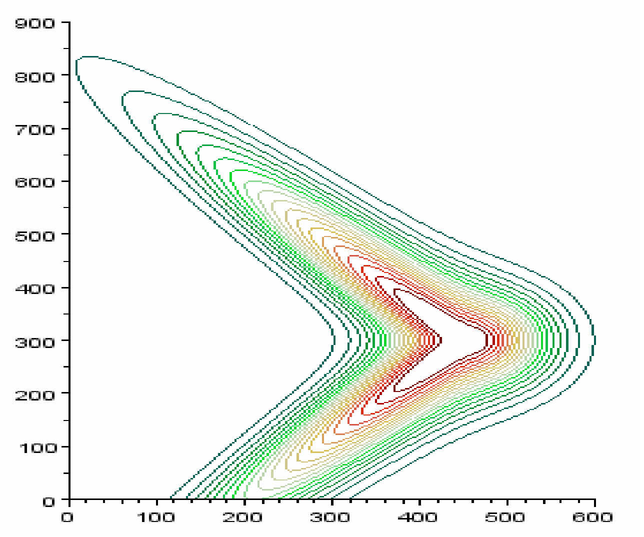


Figure 2. k=0.15, φ0=0.95, d=60, φ=φ0 sinA

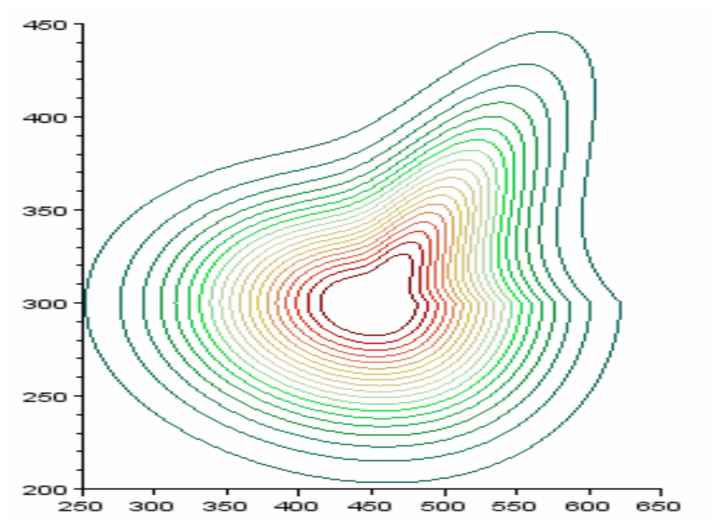


Figure 3. k=5, d=80, φ0=1, φ=φ0 ((A-π)/π)2

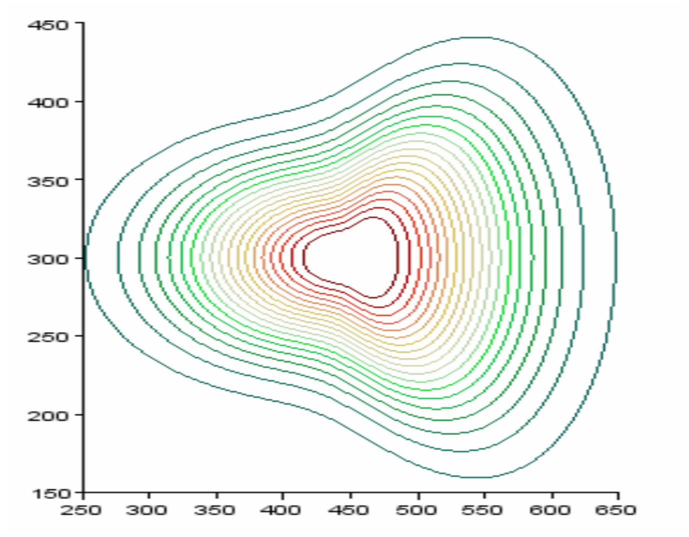


Figure 4. k=5, d=80, φ0=1.1, φ=φ0 ((A-π)/π)2 sinA

The above presented shapes are still too regular in comparison to accidental tectonic forms. The height of the real tectonic forms sometimes grows up, sometimes decreases, moreover they have many tops. In polar coordinates with pole (x_0, y_0) g function is decreasing monotonously and only by appropriate arguments transformation we may achieve multi-tops effect. Let

$v = \sqrt{x^2 + k \cdot y^2}$ means a new g function argument. This is distance from a center of the "hill". We use a following function F(v,k) as a distance transforming function:

$$F(v,k) = v \cdot \{1 - \exp[-(v - v_1)(v - v_2) - (v_1 - v_2)^2 / 4 - \\ -k \cdot \ln((v_1 + v_2)/2)]\} \tag{5}$$

The parameter k means the same elongation constant k as in (3). We see the plot of function F(v,k) on Figure 5 below.

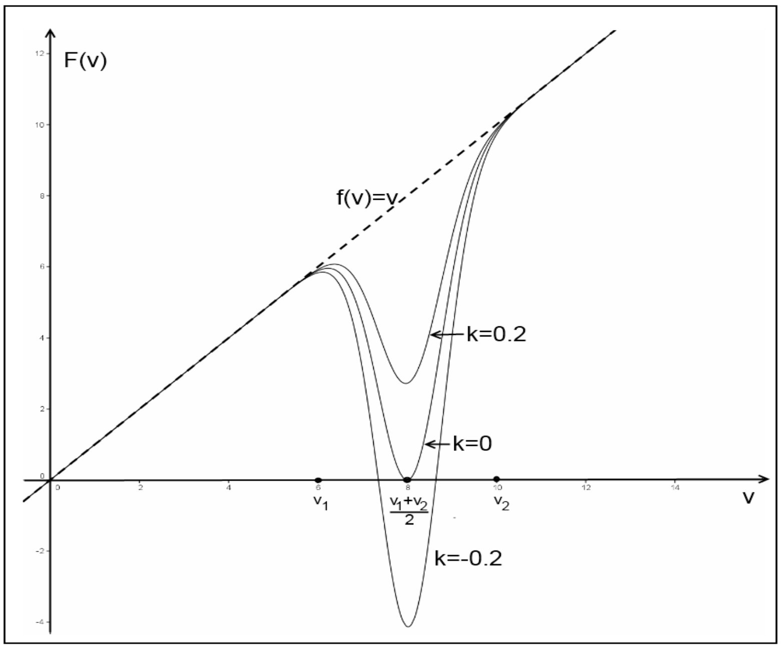


Figure 5. Plot of function F(v): layered three function for k=0 and k=±0.2. For comparison doted line is a plot of linear function f(v)=v.

This function allows to create multi-top plateaus and mountains by control k values as function of azimuth A. The average arithmetic value of few F(v,k) function allows a precise modeling of a hilly or mountainous area of the DTM (see Figure 6).

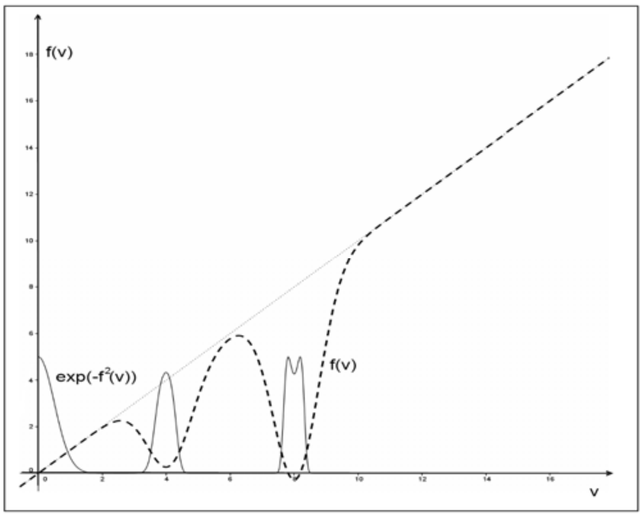


Figure 6. The way of modeling the multiple hills area.

The next step of creating the DTM is controlling a ridge direction and inclination. For our purpose two ridges of any hill are sufficient . We choose two random directions (two azimuths A1 and A2) from interval <0,2π) and two DA1, DA2 from interval < 0, π/2 >. The firsts are direction and the second broadness of a hill ridges. Because the random number generator is determined, action begins with generating a few random numbers. The parameter k is multiplied by function of azimuth fg(A), it is simply sum of two Gauss function:

$$\mathrm{fg}(A) = b \cdot \exp(-(A - A_1)^2 / DA_1^2) + \exp(-(A - A_2)^2 / DA_2^2) \tag{6}$$

This function must be periodic and continuous inside <0,2π>, to avoid fault for azimuth zero: fg(0) = fg(2π). The strict formula of function fg(A) is:

$$\begin{aligned}
&\mathrm{fg}(A) = \mathrm{fg}(p(A));\\
&p(A; A_1 < \pi) = (A - A_1)[\mathrm{sign}(A) - \mathrm{sign}(A - A_1 - \pi)]/2 -\\
&\qquad -(A - A_1 - 2\pi)[\mathrm{sign}(A - 2\pi) - \mathrm{sign}(A - A_1 - \pi)]/2 +\\
&\qquad +(2\pi - A_1)[1 - \mathrm{sign}(A)];\\
&p(A; A_1 > \pi) = (A - A_1)[\mathrm{sign}(A - A_1 + \pi) - \mathrm{sign}(A - 2\pi)]/2 +\\
&\qquad +(A - A_1 + 2\pi)[\mathrm{sign}(A) - \mathrm{sign}(A - A_1 + \pi)]/2\ ;
\end{aligned} \tag{7}$$

The plot of periodic Gauss function inside interval <0,2π> shows Figure 7

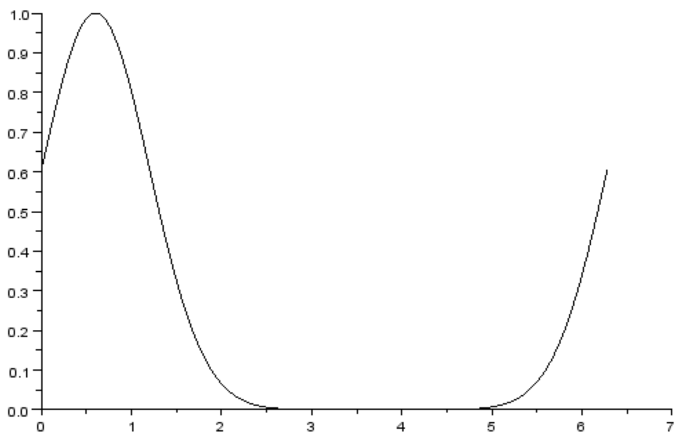


Figure7. The periodic Gauss function inside interval < 0, 2π >, A1=0.6, DA=0.6.

The composition of two fg functions controls k parameter in transformation (5) (see Figure 8 below).

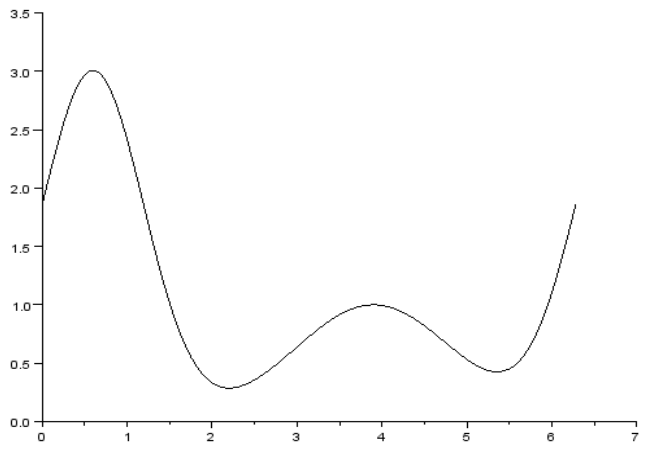


Figure 8. The sum of two periodic Gauss function inside < 0, 2π >, A1=0.6, DA1=0.6, A2=3.9, DA2=0.95, b=3.

We calculate height at point (x,y), so we need unambiguously evaluate the azimuth A at every point relative polar reference system with the center in a top of the hill. Easily to check, that function:

$$\begin{aligned} A = f(x,y) = {} & \frac{\pi}{2}\cdot \mathrm{sign}(y)\cdot[1-\mathrm{sign}(|x|)] + \mathrm{arctg}\left(\frac{y}{x}\right)\cdot \mathrm{sign}(|x|) + \\ & +\frac{\pi}{2}\cdot\{\mathrm{sign}(y)\cdot[1-\mathrm{sign}(x)]-[1-\mathrm{sign}(|y|)]\cdot[1+\mathrm{sign}(x)]- \\ & -[1-\mathrm{sign}(|x|)]\cdot[1+\mathrm{sign}(y)]\}; \end{aligned} \tag{8}$$

satisfied this conditions.

The constants v1 and v2 are multiplied by function fg(A) and next f(F(v)) is evaluated. Finally to increase impression of naturalness f(v) it is multiplied by polynomial of degree 4 of A variable:

$$z(A) = a - c\cdot A(A-A_1)(A-A_2)(A-2\pi)/(\max(z)-\min(z)) \tag{9}$$

where a and c – free constants.

5 Building a Model of the DTM.

There is shown action of the above functions in building a model of the DTM. In first step it is set a number of highlands and mountains at area of DTM. For our purpose we set four highlands with maximal height range 100-250m, two middle mountains height range 1200-1600m and one mountains with maximal height 2500-3000m. We set central coordinates of each formation, its size, elongation, initial rotation angle and v1 and v2 constants. Then the grid and scale size D is established, in this work N=4000, M=4000 and D =0,5km. So basic DTM covers area 2000x2000 km and contains 16008001 points with evaluated height. Numerical program sum up heights of all elements evaluated over all DTM area and saves it in matrix h(x,y). Figure 9 shows summarized colored contour plot of matrix h(x,y) evaluated for smaller grid size 800x600km.

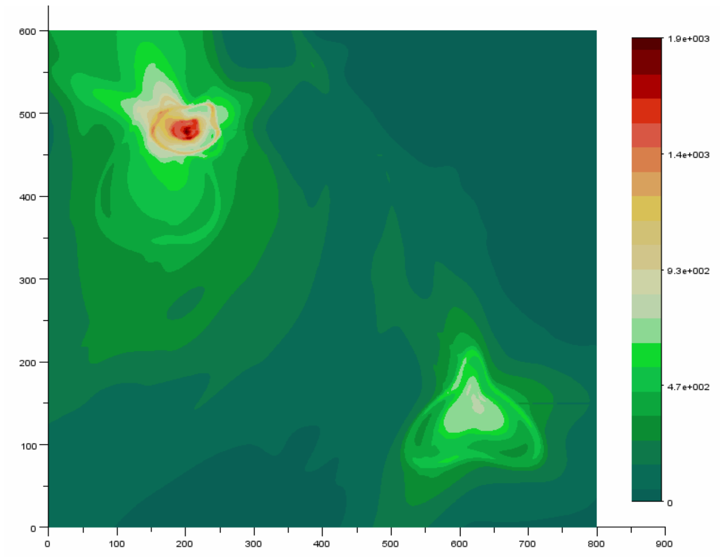


Figure 9. Summarized colored contour plot of matrix h(x,y).

Because of great scale of this map, some of fragments seems to be poor, without details. But enlarged left lower and upper right quadrant looks much better and shows appreciable diversity (Figure 10 and 11).

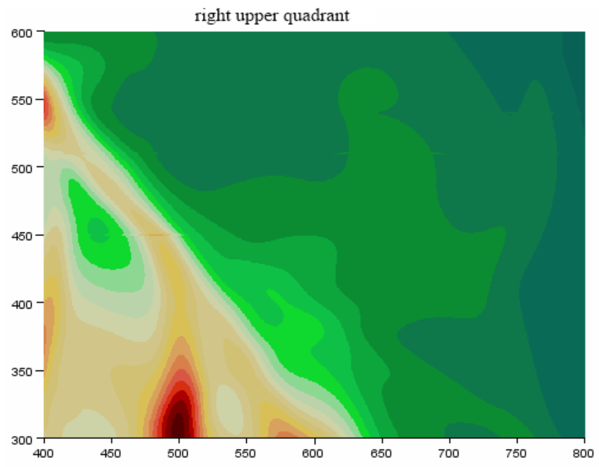


Figure 10 Enlarged upper right quadrant of the map from Figure 9

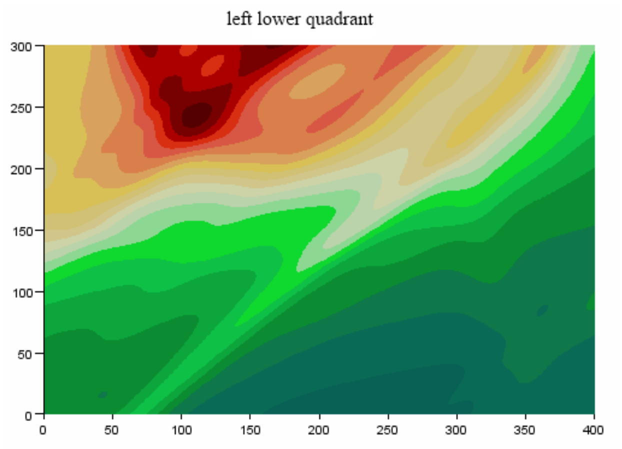


Figure 11. Enlarged left lower quadrant of the map from Figure 9.

The final sparse DTM (grid 0,5x0,5km or 16”x16”) is a sum of four above operations with the center coordinates and rotation angle randomly changed from 0 to ±50% of their initial values. The function φ(A) has been randomly chosen from three options (4) for every step of calculation.

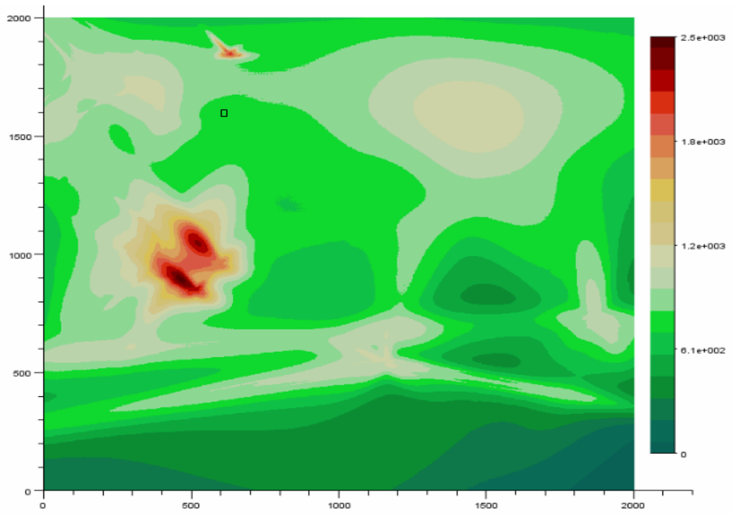


Figure 12. The colored contour plot of basic sparse height’s matrix DTM. Maximal height h=2345.95m.

The layered effect will be seen on Figure 12 above.

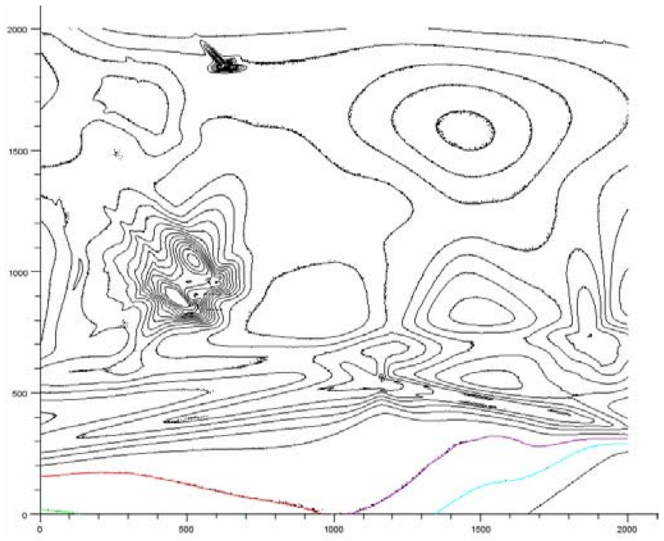


Figure 13. The contour plot of the basic sparse matrix DTM from Figure12

Below are shown the meridian profiles of terrain height.

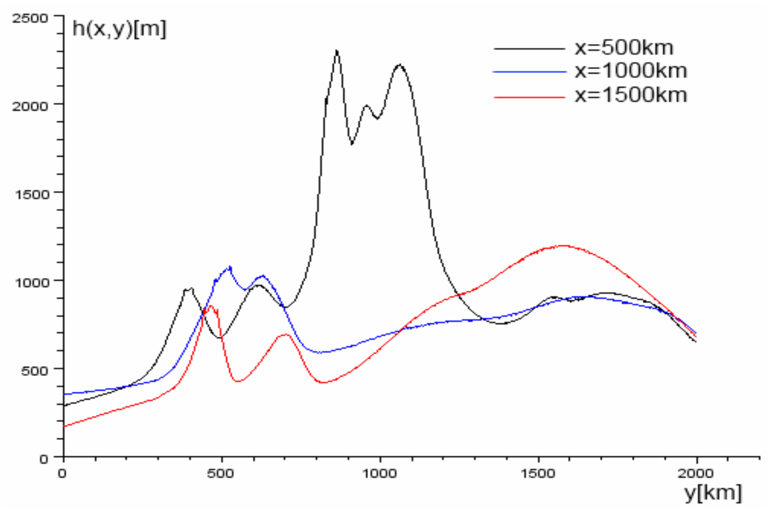


Figure14. The meridian profiles of the terrain height of the map from Figure12

Before operation of building height's matrix on the denser grid (1"x1" or 31.25mx31.25m) a local random change of height was introduced. The following algorithm has been used. A generator of random number is too predictable, so another algorithm of randomization has been used. The first five elements from two consecutive columns of pz matrix were calculated as follows:

```
    z(j+1,k)=exp(v(1,j));
    z(j+1,k)=z(j+1,k)-int(z(j+1,k));
    z(j+1,k+1)=exp(v(2,j));
    z(j+1,k+1)=z(j+1,k+1)-int(z(j+1,k+1));
```

where j=1:5 and v(2,5) is random matrix.

Next all elements of the matrix pz(4000,4000) were calculated accordingly to the algorithm:

```
    z(j,k)=exp(z(j-5,k+1));
    z(j,k)=z(j,k)-int(z(j,k));
    z(j,k+1)=exp(z(j-5,k));
    z(j,k+1)=z(j,k+1)-int(z(j,k+1));
```

j=6:4000 step 5 and k=1:4000 step 2.

All values of this matrix lies inside (0,1) interval. The new local height was calculated according to formula:

$$h = h*[1+(0.5-pz)/100]$$

so the height's changes doesn't exceed 1% of the initial value.

6 The Dense Grid and the High Resolution DTM

The base of the model of DTM is grid of points. As it was pointed out earlier, the DTM doesn't have to be completely real to main purpose. For this reason earth surface may be represented by planes(facets) defined for every three points of basic sparse grid. From a rectangular grid we must move to the triangle grid.

Simple dividing of each rectangular by diagonal is ambiguous (see Figure 15).

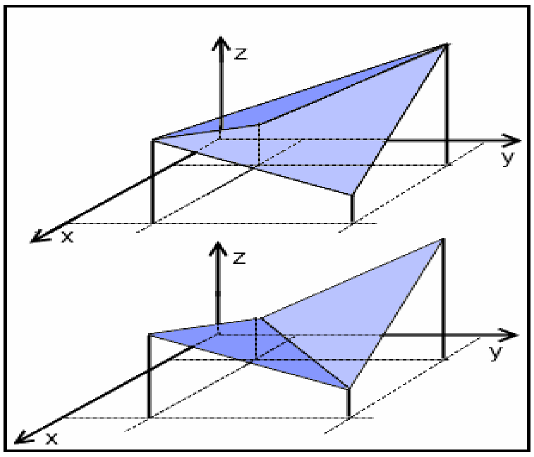


Figure 15. Two different triangle facets from rectangular grid. .

To avoid this problem, we have divided every rectangle into four triangles by cross-point of its two diagonal (Figure 16).

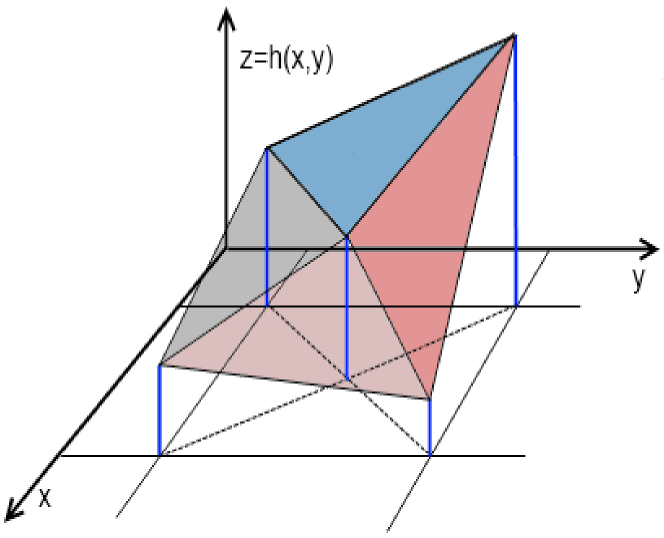


Figure16. A rectangular dividing into the triangle.

We fix the height at the cross-point (center of rectangle) as an arithmetic average of heights at rectangle apexes. The terrain surface is built from isosceles rectangular triangles (see Figure 17)

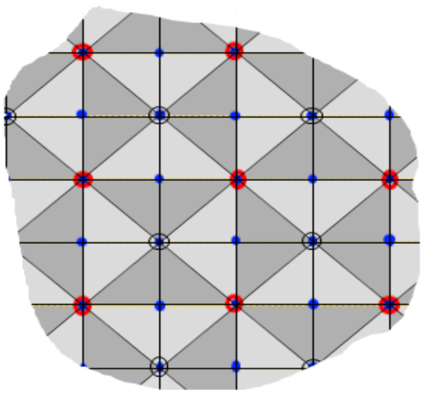


Figure 17. Part of a triangle grid. Red points belongs to sparse basic grid. Dark and bright triangles constitute area surface.

From practical point of view, the most convenient rising of map density is multiplication by 2n, then n controls density growth process. The matrix size increase 22n-time. For n=4 it means 256 time greater set to save and proportional calculating-time growth. It is wise to calculate only one time the whole DTM, save it and use, when needed.

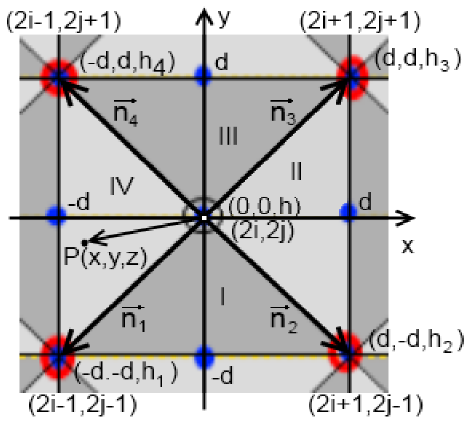


Figure18. Local coordinate system at basic sparse grid; (d,d,h) –Cartesian coordinates, (i,j)- numerical integer coordinates.

Choosing local coordinate system like on Figure18, we set four triangles with common apex and four vectors constituting that triangles: $\vec{n}_1, \vec{n}_2, \vec{n}_3$ and $\vec{n}_4$ with common initial point. A height z=h(x,y) is the third coordinate of each apex point. The coordinates of any point $\vec{r}$ inside closed area of each triangle fulfill plane equation:

$$(\vec{r}-\vec{r}_0)\cdot\vec{N}_{ij}=0; \qquad \vec{N}_{ij}=\vec{n}_i\times\vec{n}_j \qquad (10)$$

where $\vec{n}_i,\vec{n}_j$ - vectors constituting a triangle with point P(x,y,z), $\vec{r}_0=[0,0,h]$ - common points of all triangles. After short calculations we obtain required result:

$$z=h(P)=h-\frac{xN_{ij}^{x}+yN_{ij}^{y}}{2d}; \qquad (11).$$

where d=D/2, h is an arithmetic average height at center of rectangle and x,y are the coordinates on local grid:

$$\begin{cases} x=k\cdot D/2^n; \\ y=l\cdot D/2^n; \end{cases} \quad k,l\in(-2^{n-1};2^{n-1}); \qquad (12)$$

The integer coordinates of a points of the dense grid are:

$$\begin{cases} ix=(i-1)\cdot 2^2+k; \\ iy=(j-1)\cdot 2^2+k; \end{cases} \quad i\in<1;N>;\ j\in<1,M>;\ k\in<1;2^n> \qquad (13)$$

The Figure18 below shows colored contour plot of a small part of the dense high resolution DTM.

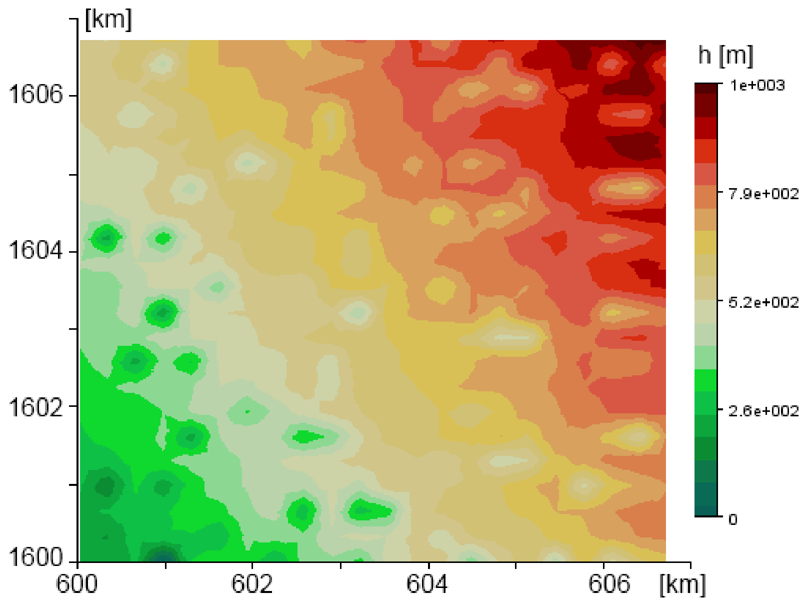


Figure19. High resolution 16-time enlarged signed fragment of the map from Figure12.

7 Summary

The model of digital terrain map was established. The destination of this model is testing some theoretical problems of earth potential and geoid determination. Starting from low resolution grid (0.5x0.5km or 16”x16”) the model containing all features of real terrain was build up mathematically. The high resolution (1”x1”) grid for testing the theoretical models of geoid determination is needed. The earth surface may be represented acceptably by triangular facets 250x500m and heights estimated at that planes. We have achieved high resolution of DTM (31x31m or 1”x1”), sufficient for numerical integration of terrain geophysical potential . The DTM features simulates natural real mass composition over earth geoid. The size of DTM is sufficient for numerical calculations of potential integrals over big areas, noted by some authors [8,9,16,17]. The model of DTM is going to be useful for testing accuracy of the methods applied by some authors to geoid determination, especially to determine the ellipsoidal correction of terrain potential [3,4,5,6,7,10,14]. The functional dependence of an ellipsoidal correction from geodetic coordinates may be established too.

References

1. Andritsanos V.D., Fotiou A., *Paschalaki E., PikridasC., Rossikopoulos* D., Tziavos I.N., 2000; Local Geoid Computation and Evaluation, Physics and Chemistry of the Earth (A), vol.25, No1, pp.63-69
2. Czarnecki K., 2010, Geodezja współczesna w zarysie, Wydawnictwo Gall, Katowice 2010.
3. Dahl O.C., Forsberg R., 1999; *Diffrent Ways to Handle Topography in Practical Geoid Determinations, Physics and Chemistry of the Earth (A)*, vol.24, No1, pp.41-46
4. Ellmann A., Vaniček P., 2007*;UNB application of Stokes-Helmert's approach to geoid computation,* Journal of Geodynamics 43, pp.200-213
5. Flury J., 2006, Short-wavelength spectral properties of the gravity field from a range of regional data sets, Journal of Geodesy 79 pp. 624-640
6. Huang J., Veronneau M., Pagiatakis S.D., 2003; On the ellipsoidal correction to the spherical Stokes solution of a gravimetric geoid, Journal of Geodesy 77, pp. 171-181.
7. Jekeli C., Serpas J.G., Review and numerical assessment of the direct topographical reduction in geoid determinations, Journal of Geodesy, pp. 226-239.
8. Kaźmierczak A., Potencjał topograficzny mas nad geoidą, in Pozyskiwanie i przetwarzanie informacji w geodezji i kartografii no 1, pp. 21-45, Akademicka Oficyna Wydawnicza EXIT, Warszawa 2012
9. Kryński J., Precyzyjne modelowanie quasigeoidy na obszarze Polski – wyniki i ocena dokładności, Instytut Geodezji i Kartografii, Warszawa 2007
10. Nowak. P., Vaniček P., Martinec Z., Veronneau M., 2001, Effects of the spherical terrain on the gravity and the geoid, Journal of Geodesy 75, pp.491-504
11. Sjöberg L.E., 2000; Topographic effects by Stokes-Helmert method of geoid and quasigeoid determinations, Journal of Geodesy 74, pp. 255-268
12. Sjöberg L.E., 2003, A computational scheme to model the geoid by the modified Stokes formula without gravity reductions, Journal of Geodesy 77, pp. 423-432.
13. Sjöberg L.E., 2004; The ellipsoidal corrections to the topographic geoid effects, Journal of Geodesy 77, pp. 804-808.
14. Sjöberg L.E., 2007; The topographic bias by analitycal continuation in physical geodesy, Journal of Geodesy 81, pp.345-350.
15. Sjöberg L.E., 2009; On the topographic bias in geoid determinations by external gravity field, Journal of Geodesy 83, pp. 967-972
16. Sun W., 2002, A formula for gravimetric terrain corrections using powers of topographic height, Journal of Geodesy 76, pp. 399-406

17. Tziavos I.N., Vergos G.S., Grigoriadis V.N., 2010; Investigation of topographic reduction and aliasing effects on gravity and geoid over Greece based on various digital terrain models, Surveys in Geophysics 31, pp. 23-67
18. Vaniček P., Nowak. P., Martinec Z., Veronneau M., 2001, Geoid, topography, and the Bourguer plate or shell, Journal of Geodesy 75, pp.210-215

CALCULATION OF STRESS-STRAIN STATE ELASTIC PLATE WITH NOTCH OF AN ARBITRARY SMOOTH CONTOUR

Ihor S. Kuz[1], Olga N. Kuz[2]

[1]Department of Mechanics and Mathematics, Ivan Franko
National University of L'viv, Ukraine

[2] Vyacheslav Chornovil Institute of Ecology, Nature Protection and Tourism,
National University, Lviv Polytechnic, Ukraine
olyakuzon@gmail.com

Abstract

The paper contains comparing calculations of the stress fields in an elastic plate with notch along the arc of a circle, ellipse or parabola obtained by analytic-numerical method based on complex Kolosov-Mushelishvili potentials and by numerical variation-difference method. These fields differ by no more than 2%, which, in particular, indicates the reliability of such numerical implementation.

Key words: semi-plane, plate, notch, variation-difference method, stress field

1 Introduction

Investigation of the stress-strain state of the plate structural elements weakened by notch, [2] is a necessary step in the calculation of their strength and reliability. Since these structural elements have finite dimensions or curvilinear boundary, the possibility of the application of analytical methods for solving the corresponding boundary value problems [4] is significantly limited, and in most cases impossible.

In this paper, we provide comparison of the obtained solutions of plane elasticity problems on uniaxial loading of a plate structural element with notch of an arbitrary smooth contour by analytic-numerical method using the complex Kolosov-Mushelishvili potentials [7] and numerical method using the variation-difference method [6].

2 Analytic-numerical method for solving the problem

Let us find the stress state of a plate of the thickness h, which is simulated by the half-plane, on the surface of which a notch is made of an arbitrary smooth contour [7]. We assume that the half-plane extends to infinity by normal stress of value P (Figure 1), and the boundary of the half-plane with notch is free from stresses.

Choose a Cartesian coordinate system Oxy, directing the axis Ox along the straight edge, and the vertical axis upward. The curve traced by the notch is denoted by L, the straight line portion of the boundary of the half-plane by L'. The lower half-plane of the plane xOy is denoted by S^-, the upper one by S^+.

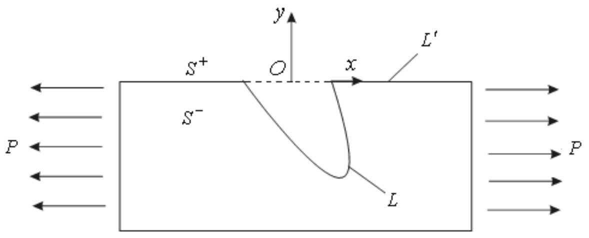


Figure 1. Plate element with notch under uniaxial loading

According to the formulation of the problem we have the following boundary conditions:

$$\sigma_{yy} = 0, \quad \sigma_{xy} = 0, \quad x \in L'; \; N = T = 0, \quad t \in L,$$

where N and T are the normal and tangential components of the vector of stresses on L respectively.

To solve the problem, we introduce the complex Kolosov-Mushelishvili potentials $\Phi(z)$ and $\Psi(z)$ [2] and present them in the form

$$\Phi(z) = \Phi_1(z) + \Phi_2(z) + \frac{p}{4}, \quad \Psi(z) = \Psi_1(z) + \Psi_2(z) - \frac{p}{2}.$$

Here $\Phi_2(z)$ and $\Psi_2(z)$ are complex potentials which are holomorphic in the lower half-plane and must ensure that the zero boundary conditions on the axis $y = 0$, are fulfilled, and corrective complex potentials $\Phi_1(z)$ and $\Psi_1(z)$ are responsible for the implementation of the boundary conditions on the surface of the notch.

Analytically extending the function $\Phi_2(z)$ from the region S^- over the region S^+ and solving the corresponding problem of linear conjugation, we obtain a singular integral equation [7], which we solve numerically using the method of mechanical quadratures [5].

3 Variation-difference method for solving the problem

We consider the plane problem of elasticity theory in a finite region V with curved boundary Σ (see Figure 1), which simulates the stress-strain state in a plate with notch of an arbitrary smooth contour. From the mathematical point of view it consists in solving equations of equilibrium in a plate [1]

$$\left(C_{ijkl}u_{k,l}\right)_{,j} = 0\,, \tag{1}$$

using mixed boundary conditions on the surface Σ

$$C_{ijkl}u_{k,l}n_j \,\big|_{\Sigma} = P_i \ . \tag{2}$$

Here C_{ijkl} are the components of the elastic modulus tensor; $u_i, P_i\ , n_j$ are the components of the displacement vector, surface forces, and the external normal to the surface Σ respectively; $u_{i,j} \equiv \partial u_i / \partial x_j$. We assume the summation from one to two by the same indices that occur twice in one expression.

For numerical solution of problem (1) - (2) it is convenient to use its variation formulation [3], which is to minimize the Lagrangian

$$L = \int\limits_V W dV - \int\limits_\Sigma P_i\ u_i d\Sigma\,, \tag{3}$$

where $W = \frac{1}{2} C_{ijkl}u_{i,j}u_{k,l}$ is the energy density of elastic deformation.

We write the Lagrangian (3) in the canonical field V_0 , which can be a rectangle or a region composed of rectangles. For this purpose we use a discrete bijection mapping of the grid in a curvilinear region V to a uniform rectangular grid $N_1 \times N_2$ of the region V_0 (Figure 2)

$$x_i = x_i(\beta^1, \beta^2) \quad (i = 1{,}2)\,, \tag{4}$$

Then $J = \det(A_i^j)$, $g_{ij} = A_i^m A_j^m$, where $A_i^j = \partial x_i / \partial \beta^j$ is the Jacobi matrix of this mapping. Using (4) we write the energy density of the deformation W in the coordinates $\vec{\beta}$

$$W = \frac{1}{2} C^{ijkl} u_{i,j} u_{k,l} = \frac{1}{2} C^{ijkl}\left(\vec{\beta}\right) B_j^m B_l^n u_{i|m} u_{k|n} = \frac{1}{2} D^{imkn}\left(\vec{\beta}\right) u_{i|m} u_{k|n}$$

where $u_{i|m} \equiv \partial u_i / \partial \beta^m$, $B_j^m = \partial \beta^m / \partial x_j$, $D^{imkn} = C^{ijkl} B_j^m B_l^n$.

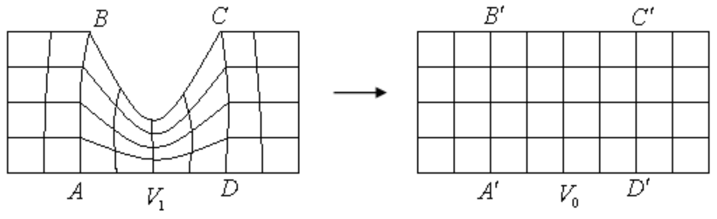


Figure 2. Mapping of a grid in the curvilinear region V onto the uniform rectangular grid in the region V_0

Thus, the Lagrangian in the rectangle V_0 will look like:

$$L_0 = \frac{1}{2} \int_{V_0} J D^{imkn} u_{i|m} u_{k|n} dv - \int_{\Sigma_0} q\left(\vec{\beta}\right) P_i u_i d\Sigma, \tag{5}$$

where $q(\vec{\beta}) = \begin{cases} \sqrt{g_{11}}, & \beta^2 = \{0, l_2\}, \\ \sqrt{g_{22}}, & \beta^1 = \{0, l_1\}. \end{cases}$

Replacing in (5) all continual function by grid ones, integrals by finite sums, and derivatives by difference derivatives, we obtain the difference analogue of the Lagrangian L_0^h using the discrete analogue of mapping (4), which should not be given analytically, in particular, to be conformal. It is sufficient to have one correspondence between nodes in the curvilinear V_1 and model V_0 regions. To determine the stationary point L_0^h we obtain a system of linear algebraic equations

$$\partial L^h / \partial v_\beta^h (i_1, i_2) = 0, \quad i_\alpha = 1,2,\ldots,N_\alpha, \quad \alpha, \beta = 1,2. \tag{6}$$

This approach leads to the impossibility of the use of direct methods for solving the system (6) due to the accumulation of errors of rounding. However, it was done with a combined iterative process that implements the scheme of the gradient method and the method with Chebyshev set of iterative parameters [8]. The complexity of its practical implementation is selection of iterative parameters.

The described variation-difference method in domains with curved boundary is implemented as a software on FORTRAN.

4 Results

For example, the calculations of the components of the stress tensor on the notch and near it done, if its boundary is an arc of the circle, ellipse or parabola.

In Figure 3 and Figure 4 there are shown the graphs of dimensionless stresses $\sigma^0_{\theta\theta} \equiv \sigma_{\theta\theta}/P, \sigma^0_{xx} \equiv \sigma_{xx}/P$ and $\sigma^0_{yy} \equiv \sigma_{yy}/P$ for the notches along the arc of the circle. Here in after, $2l$ is the width of the notch (along the axis Ox); δ is the depth of the notch (along the axis Oy); $a = \delta/l$ is a dimensionless parameter relative absorption; $\sigma^0_{\theta\theta}$ are dimensionless circumferential stresses on the notch, $\sigma^0_{xx}, \sigma^0_{yy}$ are dimensionless normal stresses on a segment $x^0 \equiv x/l = 0$, $y^0 \equiv y/\delta \in [-5,-1]$ (along the axis Oy below the groove). The hatched lines represent stress obtained by the analytic-numerical method, and the solid lines by variation-difference method.

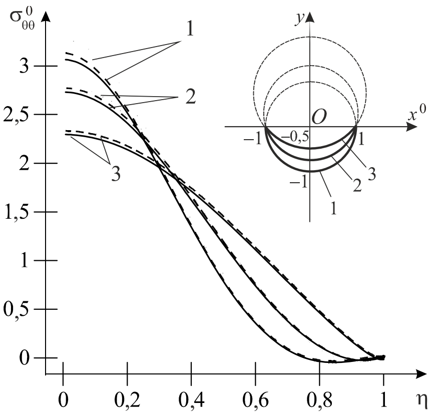


Figure 3. Stress $\sigma^0_{\theta\theta}$ on the notch in an arc of the circle for $l = 1$at different values of δ

In Figure 3, the curves *1* are constructed for $\delta = 1$, curves *2* for $\delta = 0{,}75$ curves *3* for $\delta = 0{,}5$. As seen from this figure, the stress $\sigma^0_{\theta\theta}$ (actually the coefficient of stress concentration) in the top of the notch ($\eta = 0$) achieves its greatest value in the case of notch along the semicircle (curve 1). And it is only slightly higher than typical for the Kirsch problem value 3.

Here in after, the accuracy of the results is of four significant digits (the error of about 0,1%). Monitoring convergence and accuracy of analytic-numerical and numerical solution is conducted by comparing the studied variables on the grids with single and double number of nodes.

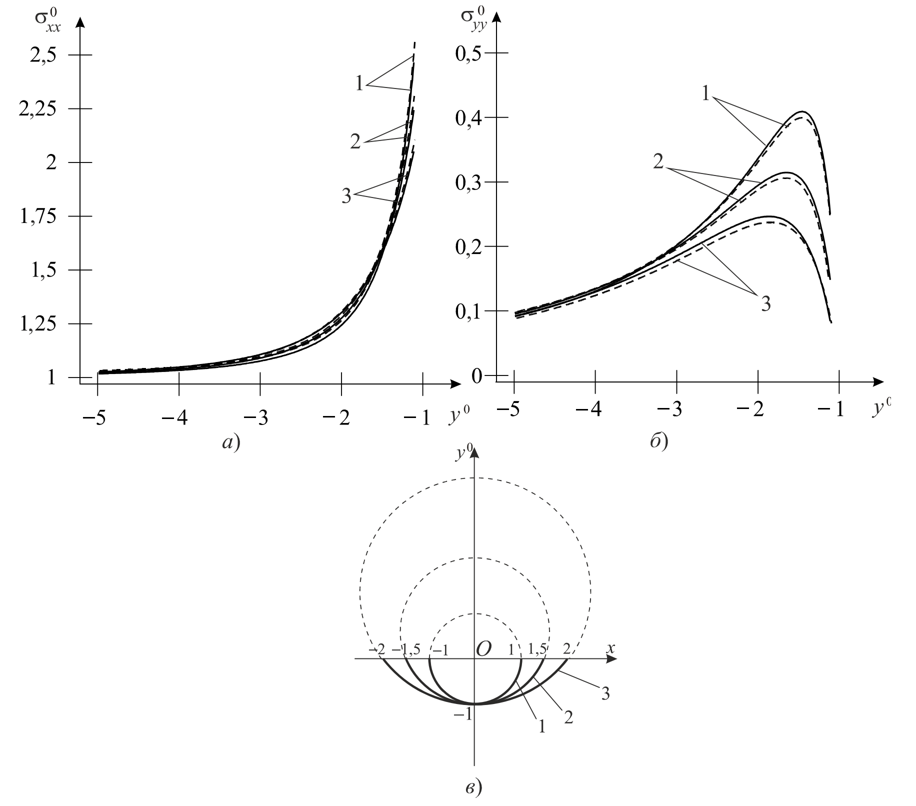


Figure 4. Stresses σ_{xx}^{0} and σ_{yy}^{0} on extension of the axis of symmetry of the notch along the arc of a circle δ=1 for various values of l.

In Figure 4, the curves *1* are constructed for $l = 1$, curves *2* for $l = 1{,}5$, curves *3* for $l = 2$. As shown in Figure 4, the normal stress σ_{xx}^{0} much lower of the notch ($y = -5$) is almost equal P , and at the top of the notch it is obvious that $\sigma_{xx}^{0} = \sigma_{\theta\theta}^{0}$.

Figure 5 and Figure 6 show the relevant graphs of stresses $\sigma_{\theta\theta}^{0}$, σ_{xx}^{0} and σ_{yy}^{0} for the notches along the arc of the ellipse.

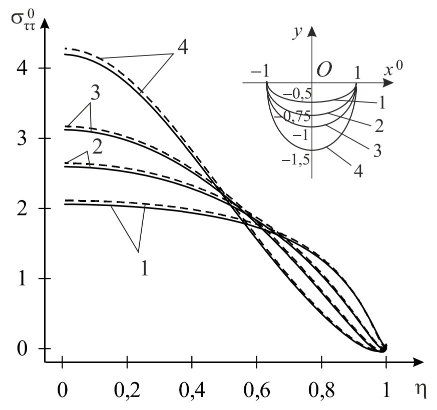


Figure 5. Stresses $\sigma^0_{\tau\tau}$on the notch along the arc of the ellipse for $l = 1$ and various values of δ.

In Figure 5, the curves *1* are constructed for $\delta = 0{,}5$, curves *2* for $\delta = 0{,}75$, curves *3* for $\delta = 1$, curves *4* for $\delta = 1{,}5$.

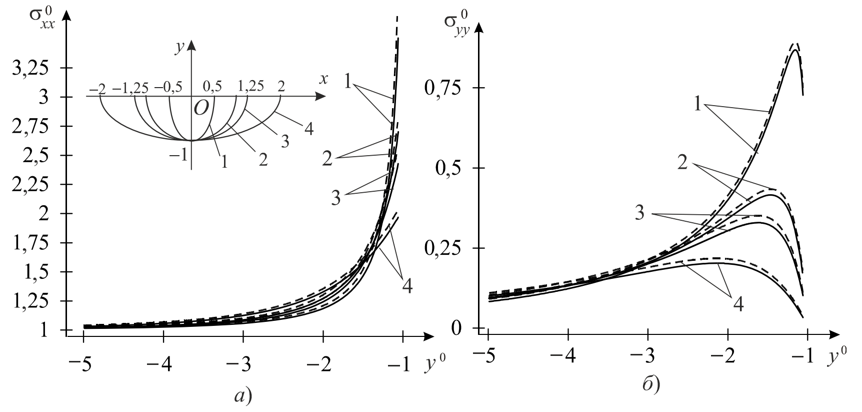


Figure 6. Stresses σ^0_{xx} and σ^0_{yy} on the extension of the symmetry axis of the notch along the arc of the ellipse for $\delta = 1$ and various values of l

In Figure 6, the curves *1* are constructed for $l = 0{,}5$, curves *2* for $l = 1$, curves *3* for $l = 1{,}25$, curves *4* for $l = 2$.

Figure 7 and Figure 8 show the relevant graphs of stresses $\sigma^0_{\tau\tau}$, σ^0_{xx} and σ^0_{yy} for the notches along the arc of the parabola.

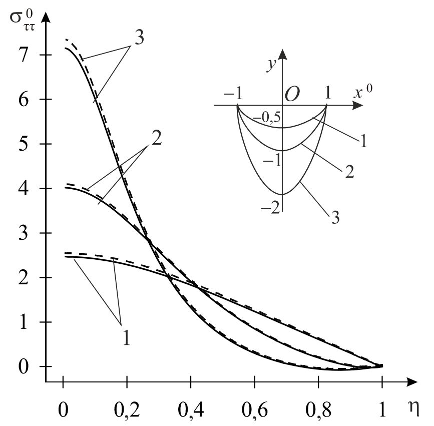


Figure 7. Stresses $\sigma^0_{\tau\tau}$ on the notch along the arc of the parabola for $l = 1$ and various values of δ

In Figure 7, the curves *1* are constructed for $\delta = 0{,}5$, curves *2* for $\delta = 1$, curves *3* for $\delta = 2$.

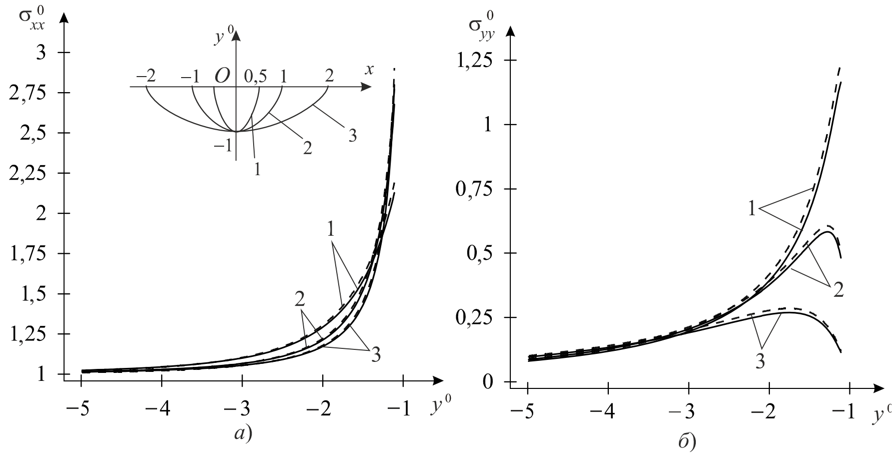


Figure 8. Stresses σ^0_{xx} and σ^0_{yy} on the extension of the symmetry axis of the notch along the arc of the parabola for $\delta = 1$ and various values of l

In Figure 8, the curves *1* are constructed for $l = 0{,}5$, curves *2* for $l = 1$, curves *3* for $l = 2$.

Figure 9 shows the change of the stresses $\sigma^0_{\tau\tau}$ for the three types of notches of the same depth (arcs of circles, parabolas and ellipses that pass through three fixed points), which enables us to identify the influence of notch shape on the stress state of the plate.

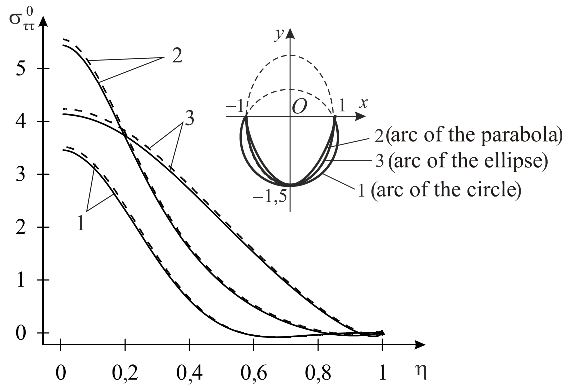


Figure 9. Stresses $\sigma^0_{\tau\tau}$ on the surface of the notches of the same depth along the arc of the circle, parabola, and ellipse for r $l = 1, \delta = 1{,}5$.

In Figure 9, curve *1* concerns the circle, curve *2* the parabola, and curve *3* the ellipse.

5 Conclusions

As is shown in Figures 5-8, the given stresses at the top of the notch along the arc of an ellipse or a parabola significantly increase with increasing of the relative depth of the notch (while increasing its depth or decreasing width). As is shown in Figure 9, sharpness of the obviously also enlarges the level of stress.

As is shown in Figures 3-9, the stress fields obtained by analytic-numerical and variation-difference methods differ by no more than 2%. This discrepancy can be explained by the fact that the analytical solution domain is unbounded, while the numerical calculation was carried out, obviously, for a finite field.

Thus, the developed method of numerical determination of stress and their concentrations agrees at solving plane elasticity problems in plates with notch.

References

1. Bozydarnik V. V., Sulim G. T., 2012, *Theory of Elasticity*, V. 1, Publishing House of Lutsk National Technical University, Lutsk (in Ukrainian).
2. Kuz I., Timar I., 2010, *Stress-strain state of elastic-plastic plates with cut or absolutely rigid inclusion,* Visnyk of Lviv University, Ser. Mech. et Math, 73, pp 148–154 (in Ukrainian).

3. Kuz O, 2005, *Stress state of semi-plane with notch under uniform extension,* Abstract of the Sixth Polish-Ukrainian Conference "Current problems of mechanics of nonhomogeneous media", Warsaw, pp. 76–77 (in Ukrainian).
4. Mushelishvili N. I., 1966, *Some Main Problems of Mathematical Theory of Elasticity*, Nauka, Moscow (in Russian).
5. Panasiuk V. V., Savruk M. P., Datsyshyn O. P., 1976, *Distribution of Stresses near Cracks in Plates and Shells*, Naukova Dumka, Kyiv (in Russian).
6. Pobedria B. E, 1981, *Numerical Methods in Theory of Elasticity and Plasticity*, Publishing House of Moscow University, Moscow (in Russian).
7. Savruk M. P., Kazberuk A., 2007, *Stress concentration near a rounded v-notch with arbitrary vertex curvature*, Acta mechanica et automatica, 1, 1, pp. 99–102 (in Polish).
8. Sheshenin S. V., Kuz I. S., 1990, *About an applied iterative methods*, Calculating mechanics of deformable solid, 1, pp. 63–74 (in Russian).

Dynamic Key Generation During a Communication Instance Over GSM

Joseph Zalaket[1], Khalil Challita[2]

[1] Holy-Spirit University of Kaslik, Faculty of Engineering,
446, Jounieh, Lebanon
josephzalaket@usek.edu.lb

[2] Notre-Dame University, Louaize, Faculty of Natural & Applied Sciences,
72, Zouk Mikael, Zouk Mosbeh, Lebanon
kchallita@ndu.edu.lb

Abstract

Mobile phone may become the protagonist of the new electronic technology. If we compare it with that of other technologies, the infiltration rate of mobile phones in the world is extremely high, both in cities than rural communities of the most of the countries. According to estimates made by the International Telecommunication Union the access to mobile networks is growing much faster than the access to Internet. This emergence has led many companies to allow new activities which were previously running strictly over the Internet to run over the mobile network such as the electronic payment. These circumstances make the security of mobile communication a priority to preserve the authentication, confidentiality and integrity of data sent between subscribers and mobile network. In this paper, we propose a dynamic key generation for the A5 GSM encryption algorithm to enforce the security and protect the transferred data. Our algorithm can be implemented over any GSM generation GSM/3G/4G.

Key words: mobile communication, encryption, GSM, A5 algorithm

1 Introduction

The use of mobile phones became vital in our everyday life. Most of the mobile phones are running through GSM (Global System for Mobile Communication). The security of the GSM is based on three main algorithms [1], [9]. A3 is an authentication algorithm, A8 is a key generation algorithm, these two algorithms are based on a hash function that takes as input a 128-bit key Ki stored in SIM card and at the network side and a random number 'RAND' of 128-bit to generate a 32-bit RESponse that will either verify the identity of the subscriber or deny it. These same inputs are given to A8 to generate a key of 64 bits that will be used to encrypt all the messages using an A5 algorithm. There are several versions of A5: A5/1, A5/2 and A5/3. The one implemented in the second generation and providing an acceptable level of security is A5/1.

Although there are several cryptographic algorithms used to provide security in GSM, this system is not completely secure since both A3/A8 algorithm and the A5/1 [10], A5/2 [14] and A5/3 [12] algorithms were broken, and the key used in ciphering was revealed sometimes in few seconds.

Trying to avoid attacks against different A5 versions we propose the dynamic generation for temporary keys that can be changed for multiple times during one communication instance. The dynamic key generation will replace the static single shared key originally stored on SIM card. Our method will require changing the architecture of SIM cards which can be a lack for its application as many users are running old cards and it is unreasoned to ask them to simply change their cards. To treat this lack we propose an algorithm that supports our extended SIM card architecture as well as the existing one.

2 Background

Works are done for making the authentication mutual from both the network and the mobile subscriber (MS) in such a way to avoid these attacks and many others [10-12].

The authors in [2] used the public key encryption to introduce a new mutual authentication method between the MS and the Visitor Resource Locator (VLR): Proxy signature.

In this method, the Home Resource Locator (HLR) will delegate the Mobile Station (MS, subscriber) the power to sign the nonce (random number) generated by the VLR and the VLR can verify the signature based on the HLR public key knowing that the MS is the one who signed. This model provides user privacy and non-repudiation features. Key management is easy since only one key (Public key of HLR) must be managed.

Authentication phase is divided into two parts: on-line authentication and off-line authentication. In the on-line authentication phase, the process requires that VLR must connect to HLR whenever MS demands authentication. However, without connecting to HLR to save authentication time and provide fault tolerance, off-line authentication is performed by VLR locally according to the parameters obtained from HLR in advance. Note that the first authentication request must be performed online and the subsequent authentication requests can be continually performed off-line.

A new protocol published by the IETF, “Extensible Authentication Protocol” or EAP-SIM, here knowing the weakness of the key used in encryption (64-bit Kc), they tried to make use of the 5 triplets allowed for the GSM network to send (RAND, SRES, Kc) to generate several RAND numbers and therefore produce more than one session key. All these keys will be then combined in one master key that is stronger which will be then used as input to the encryption algorithm [3].

As per [4], their proposed security system provides an authenticated session key distribution protocol between the authentication center AuC and the mobile station MS for every call attempt made by a MS. At the end of an authenticated session key distribution protocol the identities are mutually verified between the AUC of a Public Land Mobile Network PLMN and the Subscriber Identity Module SIM of a MS as well as the session key for call encryption is distributed to the MS.

A self-concealing mechanism is introduced in [5], the purpose here is to reduce the entries of the GSM databases (VLR and HLR) by discarding the bulky database and create valuable improvements for portable communication systems. In this approach, the shared secret is concealed by the authentication server and only the authentication server has the private key to open the shared secret.

The new concept initiates several positive changes. First, the sensitive and large database can be discarded. Consequently, this prevents hacker attacks to the database and reduces maintenance demand for the server.

A warrant is used to guarantee the user's access rights; an issue is not addressed in the conventional challenge-response scheme.

When it comes to our protocol, we profited from new variables that already exist in the GSM and used but not in the security algorithms such as TMSI, LAC and a new random number.

3 Dynamic key generation

Note that, any change arising in the algorithms currently implemented on the SIM card requires the generation of new SIM cards for all users and that could be inconvenient for many of them. That's why the protocol we are proposing will support two versions of SIM cards, the old one that is currently used and therefore, the subscribers that don't want to change their SIMs will keep benefiting from the security features already provided by the GSM while the holders of the new generated SIMs will profit from better security described in the following sections.

3.1 Key generation process

GSM network sends a 128-bit RAND number to the MS, they both apply the A3 (authentication algorithm) that will accept this RAND and a 128-bit key Ki stored in the SIM and known to the GSM as input to generate a 32-bit RESponse number. The MS sends this RES to the GSM network that will compare it to the response number it generated, if they match, the user is authenticated and they can now apply the A8 algorithm that takes Ki and RAND as input and generates the 64-bit key Kc that will be used in the ciphering algorithm A5.

Now that the Kc is generated, new core elements take part in the new protocol: 32-bit TMSI, 64-bit IMSI, 16-bit LAC and a new number NEWRAND that is randomly generated by the GSM.

A GSM conversation is sent as a sequence of frames every 4.6 millisecond [6] and therefore we chose to generate the NEWRAND every 10 x 4.6 ms that means every 46 milliseconds.

Once this new random number is generated, the timestamp (dd/mm/hh/mm/ss) will be divided by the NEWRAND.

This time variable has a 24-hour format to avoid having the same timestamp twice a day. A random interval was chosen to decide whether or not to use this NEWRAND. If the division result was within the interval then the NEWRAND is sent for use by both the MS and the GSM otherwise, the old value of NEWRAND is used. This interval can be left for the operator to decide its boundaries for better security and to avoid having any pattern. But in our simulation the range of values was chosen to be within [20790, 977890].

Once the NEWRAND is generated and sent, the main function called: KeyGen() will use it with other parameters to generate an enhanced security key.

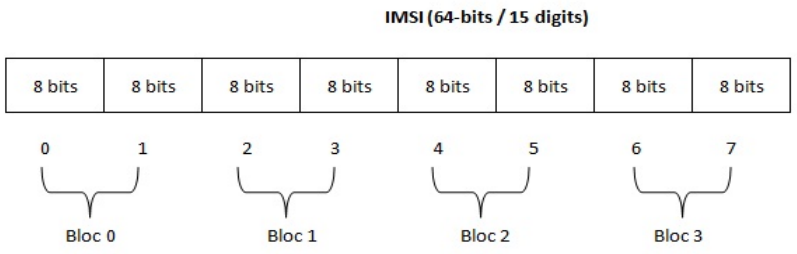


Figure 1. IMSI blocks

The main function KeyGen() takes seven parameters: TMSI, LAC, IMSI, NEWRAND, Kc, version, NewKeyGenerated.

- TMSI is the 32-bit or 8 digits value that is changing on VLR change or if the interval of time set for it expires.
- 16-bit LAC or 4 digits, location area code that will be updated at each location change. A Location Update Procedure is triggered.
- IMSI is 15 digits number (64-bits); it is the main identifier for the MS. In our protocol we will divide it (logically) into 4 blocks and based on the

NEWRAND we will select which block can be used to generate the new key as shown in Figure 1.

- NEWRAND already generated as described in the previous subsection
- Kc is the original 64-bit key generated using the A8 algorithm.
- Version is set to 1 in this protocol because it is the new version of SIM card, for the previous cards it will be set to 0. This function will only run if the version was 1.
- NewKeyGenerated is the new 64-bit key generated at each execution of the function KeyGen() and that will be used as input to the A5 ciphering algorithm.

All the values used as identifiers such as TMSI, LAC, IMSI and the key are originally sent in hexadecimal format [7]. So each one of them is converted to binary format and stored in a vector of bytes.Example: TMSI is of 8 hexadecimal digits: A1 09 34 E7, it will be converted to binary and stored in a 4 bytes vector as in Figure 2:

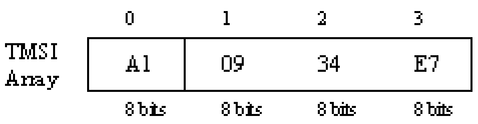


Figure 2. TMSI array

So, in memory A1 will be stored as (10100001). Same applies for all these variables in term of conversion from hexadecimal to binary.

As we can see, TMSI is an array consisting of 4 cells. The possibilities of arranging these cells in different order are 4! = 24. For this purpose, we have created a static matrix that consists of 24 rows and 4 columns. In each column is stored an index from 0 to 3. This matrix plays the role of an index to the TMSI array in order to accelerate the generation.

The remainder of the division of NEWRAND by 23 will determine the index of the row of the matrix to be chosen. Let's say the NEWRAND modulo 23 gave 5 as a result. Looking at the row of index 5, the columns contain respectively 0,3,1,2. Therefore the rearranged TMSI array will have as elements: TMSI[0], TMSI[3], TMSI[1] and TMSI[2] respectively. And thus the rearranged TMSI will be A1 E7 09 34.

The remainder of the division of NEWRAND by 23 will determine the index of the row of the matrix to be chosen. Let's say the NEWRAND modulo 23 gave 5 as a result. Looking at the row of index 5, the columns contain respectively 0,3,1,2. Therefore the rearranged TMSI array will have as elements: TMSI[0], TMSI[3], TMSI[1] and TMSI[2] respectively. And thus the rearranged TMSI will be A1 E7 09 34.

The LAC array is of 16 bits (2 hexadecimal digits), for example: F2 56.

The IMSI is divided into 4 blocks of 16 bits. Also based on this NEWRAND we will decide which block to use. The remainder of NEWRAND divided by 4 will determine the number of the block. If NEWRAND modulo 4 gave 2 as remainder, the block of index 2 of IMSI is chosen which means the IMSI[4] and IMSI[5] are used.

A new array of 4 bytes is introduced, which is the combination of the LAC array and the two chosen cells of the IMSI. This 32-bit array is XORed bit by bit with the rearranged TMSI array.

Since NEWRAND is even, the New Output will be swapped with block 1 of Kc and therefore the new key generated consists of the block 0 of Kc concatenated with the new output (32-bits) as described in Figure 3.

This whole process is repeated:

- Every 46 milliseconds
- Whenever the TMSI value is updated
- Once a Location Update Procedure is taking place

So we have multiple keys that will be generated every while. Each time a new key is generated, the A5 will call its main function using the newly generated key as input.

3.2 New algorithm for supporting existent SIM cards

Now that the multiple keys generation is solved, we still have to make the new protocol compatible with both versions of SIMs. This must be taken into consideration from the GSM network view given that it would be the one responsible of generating the NEWRAND and sending it to the MS. Since the implementation of the global function grouping all three algorithms (A3, A8, A5) is not made clear by GSM community, the following is a theory of how the protocol should be put into practice

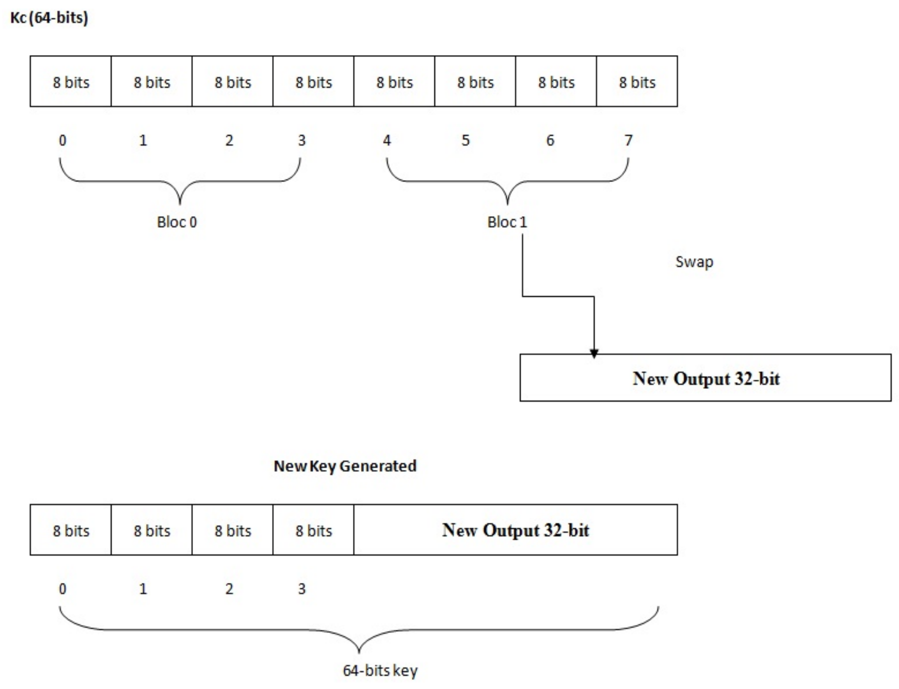


Figure 3. New Key generation

We assume that the major function that combine all the elements is a function called F() having several parameters (key, TMSI, LAC, IMSI). So if the values of the last three parameters are not sent then we know that we are using the old version of SIM cards and therefore the usual protocols will be followed. If these values are sent as parameter arguments then we are using a new version of SIM cards and therefore the KeyGen() function early described will be applied.

The below pseudo code explains the idea and how default value of null will be assigned to non sent parameters in order to be compatible with the current SIMs:

```
F(char Key,char TMSI=Null,char LAC=Null,char IM-
SI=Null)

{       If no values are set for TMSI, LAC and IMSI

        Use the old protocol and algorithms

        e.g. call g (key)

        Else
Use A3/A8 and then call the new function KeyGen

e.g. call h (Key, TMSI, LAC, IMSI)

}
```

3.3 New SIM card architecture

Further to the functions that are newly embedded in the system, we need to take into consideration the elements that were added and were not available in the previous version. The NEWRAND for example is used in our new protocol and therefore we need to store it in the SIM. For this purpose, the new SIMs will hold a new register for the NEWRAND value; each new value will override the previous one.

The LAI value is updated using a BCCH (broadcast control channel) with ongoing conversation, for this reason we will be using this same channel to send the NEWRAND whenever it is generated and the decision was to send it.

The function that is responsible of the NEWRAND generation and sending will be implemented on the network side only, while the function executed for the key generation should be available on both SIM and network.

4 Implementation

A version of our algorithm is implemented using the C language. The simulation has been done on a PC having the following specifications:

Intel Core i3 CPU, 2.40 GHz - 3 GB of DDR2 RAM - Windows 7 32-bit OS - 500 GB Hard Disk

Note that the capacity of the used machine is not important as the algorithm uses trivial system resources such that: only 30 bytes of RAM in addition to which is used by the A5/1 algorithm, 2 bytes of physical memory (register to store the random number on the SIM card side) and about 0.001ms of CPU time to generate a new key. Even though, a new implementation of this prototype is currently prepared to run on mobile phones using Android OS.

A simplified simulation of an ongoing call running during less than 0.5 second gives the following observations:

– Eight new keys have been generated during this short and the same key has been sent twice during two different sent sessions.
– The same timestamp has been assigned for all these tasks as they are done during less than one second (the necessary time to get a new timestamp).

5 Discussion

Using a single key during a whole mobile communication or even for a long period of time made it easy for cryptanalyzers to reveal the key and get all the information shared between the MS and the network. Whereas in our protocol we have more than one key that is being used within the same conversation and its way of generation is not patterned. The location update depends on whether the user is moving or not, the TMSI change depends also on the network implementation in addition the random number periodically generated is not always used so an eavesdropper might have to wait a long period of time to get the random used.

The timestamp varies every second, and its XOR Result with the NEWRAND cannot be detected since it is done on the network side. Also the range in which this result is tested can be set by the operators making it trickier for attackers. The generation of the number is done randomly so guessing this number is practically very difficult to achieve.

As known, weakness in GSM was that algorithms were never published, and therefore their security level was not evaluated. Even if our protocol is published, the key element that is changing is the NEWRAND number. So an attacker has to detect the key that was originally generated and then to track every random number that is being sent irregularly. In case he misses one of the random numbers, he won’t be able to decrypt the message.

The NEWRAND number is generated every 46 ms but the time execution of the algorithms is still the same because they are not modified. So in case we decided to generate this random less frequently (make time more than 46 ms), catching it becomes harder while generating this number in less than 46ms will make it simpler.

This new feature we offer is not mandatory; we are not forcing the users to change their SIM cards. They can keep the old ones and benefit from the same security features already offered by GSM. For those who are seeking a better level of security will choose the new version of SIM cards which can also be offered by default for new subscribers.

6 Conclusion

The GSM community pretends that its used algorithms are safe and secure. By contrast, many successful attacks have been done on each of these algorithms and this is due in the first place to the fact that the implementation of these algorithms were kept secret and they were reversed engineered.

Several attempts were made to enhance security algorithms and several propositions were made for both authentication and encryption algorithms.

We, in our turn, have proposed a new protocol for generating multiple keys in the same conversation, so each stream might be encrypted using a different session key than the previous one. The strength of our protocol lies in the fact that we depend on several variables that are very hard to be detected or guessed by an eavesdropper, the encrypted TMSI, the LAC changing at each location change and a random generated by the network but that is not constantly sent to avoid any pattern. Also the new protocol supports two versions of SIM cards. The subscribers holding the old versions will profit from the usual security features while those using the new version will benefit from a higher level of security.

We have simulated these functions on a normal PC using the C programming language, the same used in the other algorithms implementation.

Our strategy was not to hide the proposed algorithm from cryptanalyzers, but rather to use a combination of dynamic parameters that render the task of attackers semi-impossible. (e.g. the technique of public key or asymmetric key cryptography)

The A5/1 and A5/2 algorithms were breached, and the key (Kc) used in ciphering was revealed. To defend, GSM engineers implemented a 3rd generation system, but also kept secret. They claim enhancing the security authentication with their new Authentication and Key Agreement protocol (AKA) also with the new block cipher algorithm A5/3 [8] based on KASUMI which has been also attacked [12], [13].

References

1. Brakan E., Biham E. and Keller N., 2003, *Instant Ciphertext-only cryptanalysis of GSM encrypted communication*, CRYPTO 2003: 600-616
2. Haverinen H., Nokia Ed. and Salowey J., Ed. Cisco Systems, 2006, *RFC 4186 "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)"*, IETF
3. Duraiappan C., Zheng Y., *Enhancing Security in GSM*, University of Wollongong
4. Wei-Bin Lee and Chang-Kuo Yeh, 2008, *A Self-Concealing Mechanism for Authentication of Portable Communication Systems*, International Journal of Network Security, Vol.6, No.3, PP.285–290
5. Antipolis S., 2003, *Identity protection using P-TMSI for GPP/WLAN interworking*, 3GPP,TSG,SA WG3 Security – S3#26
6. Biryukov A., Shamir A., Wagner D., 1999, *Real Time Cryptanalysis of A5/1 on a PC*
7. http://www.gsm-security.net/papers/a51.shtml
8. GSM 2000 Joint GSMA TSG SA WG3 Working party, *Requirements Specification for the GSM A5/3 Encryption Algorithm, version 0.5*

9. Barkan E., Biham E., Keller N., 2008, *Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication,* J. Cryptology 21(3): 392-429
10. Barkan E., Biham E., 2002, *Conditional Estimators: An Effective Attack on A5/1. Selected Areas in Cryptography*: 1-19
11. Ekdahl P., Johansson T., 2003, *Another attack on A5/1. IEEE Transactions on Information Theory* 49(1): 284-289
12. Dunkelman O., Keller N., Shamir A., 2010, *A Practical-Time Attack on the A5/3 Cryptosystem Used in Third Generation GSM Telephony.* IACR Cryptology ePrint Archive 2010
13. Dunkelman O., Keller N., Shamir A., 2010, *A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony*. CRYPTO 2010: 393-410
14. Paglieri N., Benjamin O., 2011, *Implementation and performance analysis of Barkan, Biham and Keller's attack on A5/2,* Ensimag- Grenoble Institute of Technology - INP

DOCUMENTS PROCESSING IN THE REQUIREMENTS MANAGEMENT SYSTEM

Tomasz Wydra[1], Konrad Grzanek[2]

[1]Academy of Management, Łódź, Poland
tomasz.wydra@hotmail.com

[2] IT Institute, Academy of Management, Łódź, Poland
kgrzanek@spoleczna.pl, kongra@gmail.com

Abstract

Requirements analysis is a highly critical step in software life-cycle. Our solution to the problem of managing requirements is an embedded domain-specific language with Clojure playing the role of the host language. The requirements are placed directly in the source code, but there is an impedance mismatch between the compilation units and electronic documents, that are the major carriers of requirements information. This paper presents a coverage for this problem.

Key words: Requirements engineering, knowledge discovery, documents processing, PDF

1 Source-code Oriented Requirements Management

According to surveys (like [1]) on software quality, software development is still more art than science. Researches point out poor requirements management as one of the most important factors when discussing the possible reasons. Requirements analysis is a highly critical step in software life-cycle [2], [3]. The proper and effective requirements management saves the overall project costs due to the following reasons:

- Requirement errors typically cost well over 10 times more to repair than other errors.
- Requirement errors typically comprise over 40% of all errors in a software project.
- Small reductions in the number of requirement errors pay big dividends in avoided rework costs and schedule delays.

Moreover, the requirement managements errors are the most common errors in software projects.

Our previous work [3] describes a requirements management system for the Clojure programming language [4]. The system allows a programmer to put the requirements directly into the source code and to manage them using standard compilation techniques as well as doing it interactively using Lisp REPL (Read-Eval-Print Loop). Our unique approach is inspired by a homoiconicity of the languages from the Lisp family of programming languages (as stated in [3]). Our solution is an embedded domain-specific language with Clojure playing the role of the host language. This DSL wins the following for the analysts, designers and programmers (after [3]):

- Editing source code is a primary activity every programmer undertakes on every work-day. Putting the act of reading/writing the requirements into source code increases the comfort of this – sometimes boring – activity.
- It also affects the designers and other people not involved directly in the implementation phase, because it opens an effective channel of communication between – for instance – a system analyst and a coder; the analyst writes a requirement directly in a compilation unit, the programmer reads it and perform further steps to gain the required functionality.
- The presence of requirements in compilation units allows to interweave them (their definitions formally speaking) with source code snippets being their direct implementations or implementation parts. This point is especially important because an act of locating requirements in pure (not instrumented with requirements or requirement-related tags) source code is a tedious and hard to solve problem. Further works on this can be found in [5, 6].
- A compilation unit keeping some requirements may be tracked and managed by a source management and revision control system, such as Git [7]. An immediate consequence is the ability to manage the requirements versions, because a requirement change is a change in the compilation unit. All version control system's goodies, including the possible encryption and the overall robustness of a distributed versioning system are there to be used.

2 The Requirements and Implementations Abstraction

The model of the requirements management system presented here consists of two major types of data called ***REQ-infos*** and ***IMPL-infos***. Objects of those types are essentially maps with the properties as specified in the following tables:

Table 1. REQ-info properties

:label	A label of the requirement.
:doc	A documentation part of the requirement describing some feature or presenting some fact.
:file	A name of the compilation unit (source file) in which the requirement was defined.
:ns	A Clojure name-space in which the requirement was defined.
:line	The line number in ***:file*** in which the requirement was defined.
:score	A search score. The property is present only in the requirement records retrieved via querying.

Table 2. IMPL-info properties

:label	A label of the requirement for which the implementation was defined.
:file	A name of the compilation unit (source file) in which the implementation was defined.
:ns	A Clojure name-space in which the implementation was defined.
:line	The line number in :file in which the implementation was defined.
:score	A search score. The property is present only in the implementation records retrieved via querying[1].

REQ-info objects (***REQ-infos***) represent requirements and the ***IMPL-infos*** – the implementations. Mutual relations between the objects of both kinds form a graph. The graph may undergo further processing and searching to build a more comprehensive overview of what has to be implemented and what has already been achieved, especially when the number of requirements reaches thousands (not even mentioning number of ***IMPL-infos*** in such case).

1 Indexing and querying the REQ-infos and IMPL-infos is performed using the Lucene text search engine. More on this can be found in [3].

The relations between REQ-infos and IMPL-infos are textual and – at least by now – cannot be created fully automatically. The most important notion is a ***label***, a textual identifier representing the ***REQ-info***.

3 Labels and References

To define a ***REQ-info*** inside a source file one has to use a `(defr …)` form that comes with the library. It's list of arguments comes as follows:

```
[doc & {:keys [labels refs do]
        :or   {labels [] refs [] do []}
        :as   options}]
```

The ***doc*** argument is the textual content of a requirement, a documentation part most of the time. ***do*** is a collection of actions to perform on the compile-time, when processing the ***defr*** macro instance, with the (probably) most important ***persist*** action. An example use can be seen below[2]:

```
(REQ3/defr
  "4.1 The Kinds of Types and Values

  There are two kinds of types in the Java programming language:
  primitive types (§4.2) and reference types (§4.3). There are,
  correspondingly, two kinds of data values that can be stored
  in variables, passed as arguments, returned by methods, and
 operated on: primitive values (§4.2) and reference values
 (§4.3).

     Type:
             PrimitiveType
             ReferenceType"

  :do [REQ/persist])
```

The example above is a requirement that describes a tiny part of Java type-system coming from The Java Language Specification, Third Edition [8]. In this case a ***label*** is automatically generated for a ***REQ-info***, but in general a programmer or a system engineer is allowed to pass a vector of custom labels. This has the following desirable consequences:
- A label is an identity of a requirement.

2 The REQ library was a subject of substantial changes since it's version described in [3]. This is why the form has a different shape than the ones in the mentioned paper.

3 Possible assuming a form `(:require [kongra.req :as REQ])` was evaluated earlier in the name-space header.

– A ***REQ-info*** may be a description of a whole bunch of requirements, when it contains a number of labels larger than one.
– A requirement may be represented by a collection of REQ-infos and spread all over a large number of places in the code, possibly separate compilation units.
– So the cardinality of a relation between a requirements and ***REQ-infos*** is many-to-many.

Another important, but not explicit property of the ***REQ-info*** is it's collection of references. The ***references*** are labels of the other ***REQ-infos*** referenced from the defined one. The form `(`***`defr`***` …)` possesses a separate parameter called *refs*, but it does not map directly to a ***REQ-info*** property. Instead the REQ library contains some indirect containers for references avoiding an unnecessary *Lucene* indexation of references when storing ***REQ-infos***[4].

Labels and textual content (***doc***) together with ***references*** form a complete graph of informally formulated requirements. Informally because the portions of information are highly human-dependent and their nature does not undergo any regulations other than the syntactic ones. One way to get closer to some automatism in this regard is to:

– Allow the system to generate the labels – a default behavior of the (defr…) form.
– Use the NLP tools to extract dictionary words (1-grams) and then 2-grams, 3-grams and – in general – N-grams out of the doc part of ***REQ-infos*** to build collections of **refs**.

Realization of the two postulates will be a subject of some further work.

4 PDF Processing and the Extraction of Textual Content

Although the system allows and encourages the software engineers to put all requirements (and implementations) information into the compilation units, initially these portions of textual content are placed in some kind of electronic documents, with the PDF format being the most widely used. This section describes one possible solution for the problem of format incompatibility arising between Clojure compilation units and PDF documents. To be more precise here, it presents a way to transform an example portion of textual information present in a document into a shape acceptable from the point of view of creating a `(`***`defr`***`…)` form in a compilation unit.

We have chosen a PDF document processing library for Java called ***iText*** [9]. This library can include extracting information from the document. It

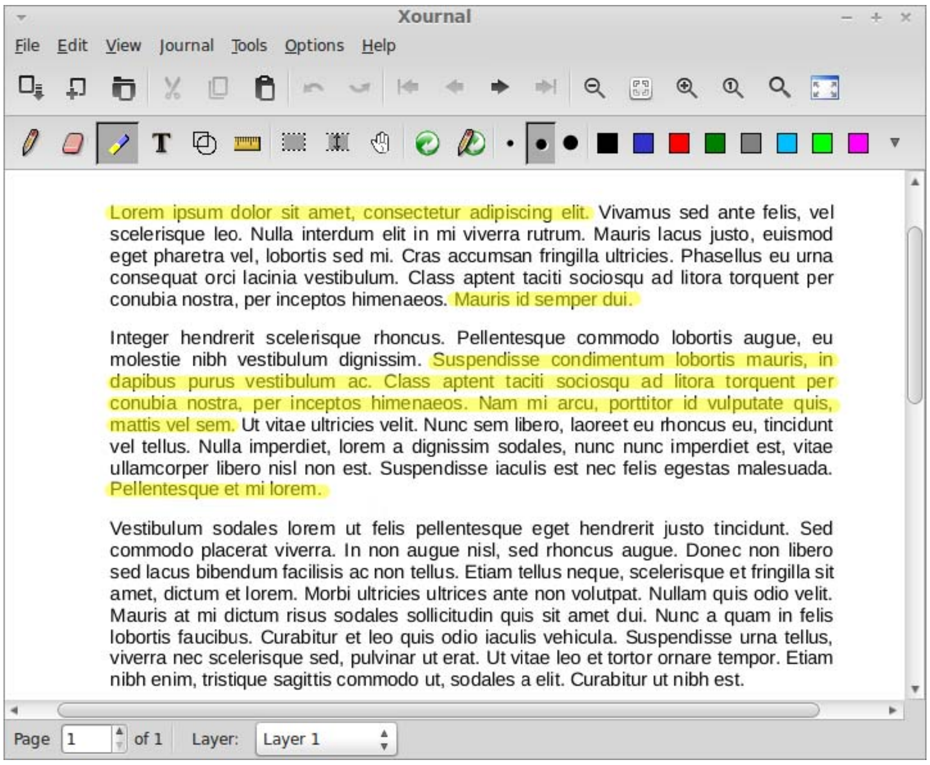
4 This could also be achieved by configuring Lucene. In one way or the other REQ treats references as a kind of the second-class citizens (derivatives) when talking about ***REQ-infos*** model.

operates on the so-called tokens that are tagged to the types of stored information. The following table provides a list of tokens which iText operates on [10]:

Table 3. Overview of the token types

Token type	**Symbol**	**Description**
NUMBER		The current token is a number.
STRING	( )	**The current token is a string.**
NAME	/	**The current token is a name.**
COMMENT	%	**The current token is a comment.**
START_ARRAY	[	**The current token starts an array.**
END_ARRAY	]	The current token ends an array.
START_DIC	<<	The current token starts a dictionary.
END_DIC	>>	**The current token ends a dictionary.**
REF	R	**The current token ends a reference.**
OTHER		The current token is probably an operator.
ENDOFFILE		There are no more tokens.

The technique used to process the pdf document is to check token by token for the specified pattern. The Xournal editor [11] was used to highlight data to be extracted either as a ***REQ-info doc***, ***label(s)*** or ***ref(s)***, whith the tool called highlighter. Sample selection can be seen at the following Figure 1.


Xournal
File Edit View Journal Tools Options Help
Page 1 of 1 Layer: Layer 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed ante felis, vel scelerisque leo. Nulla interdum elit in mi viverra rutrum. Mauris lacus justo, euismod eget pharetra vel, lobortis sed mi. Cras accumsan fringilla ultricies. Phasellus eu urna consequat orci lacinia vestibulum. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Mauris id semper dui.

Integer hendrerit scelerisque rhoncus. Pellentesque commodo lobortis augue, eu molestie nibh vestibulum dignissim. Suspendisse condimentum lobortis mauris, in dapibus purus vestibulum ac. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nam mi arcu, porttitor id vulputate quis, mattis vel sem. Ut vitae ultricies velit. Nunc sem libero, laoreet eu rhoncus eu, tincidunt vel tellus. Nulla imperdiet, lorem a dignissim sodales, nunc nunc imperdiet est, vitae ullamcorper libero nisl non est. Suspendisse iaculis est nec felis egestas malesuada. Pellentesque et mi lorem.

Vestibulum sodales lorem ut felis pellentesque eget hendrerit justo tincidunt. Sed commodo placerat viverra. In non augue nisl, sed rhoncus augue. Donec non libero sed lacus bibendum facilisis ac non tellus. Etiam tellus neque, scelerisque et fringilla sit amet, dictum et lorem. Morbi ultricies ultrices ante non volutpat. Nullam quis odio velit. Mauris at mi dictum risus sodales sollicitudin quis sit amet dui. Nunc a quam in felis lobortis faucibus. Curabitur et leo quis odio iaculis vehicula. Suspendisse urna tellus, viverra nec scelerisque sed, pulvinar ut erat. Ut vitae leo et tortor ornare tempor. Etiam nibh enim, tristique sagittis commodo ut, sodales a elit. Curabitur ut nibh est.

Figure 1. Text highlighted in Xournal editor.

Figure pattern (selection) that editor creates is shown below (1):

$$\text{gs}\ \ x_1\, y_1\ \ \text{m}\ \ x_2\, y_2\ \ \text{l}\ \ x_3\, y_3\ \ \text{l}\ \ldots\ \text{l}\ \ x_n\, y_n\ \ \text{l S} \qquad (1)$$

where:

gs – represents the beginning of the figure
x, y – coordinates of subsequent elements of figure
m, l – tokens specifying circumscribe

The algorithm realizing searching the document is shown at Listing 1.

```
for (int i = 1; i <= pageNumber; i++) {
  byte[] pageBytes = reader.getPageContent(i);
  PRTokeniser tokeniser = new PRTokeniser(pageBytes);
  TokenType tokenType;
  String tokenValue = null;
  int startOfShape = 0;      // 0 = shape not founded,
                             // 1 = the pointer passed
                             //     start of shape
```

```
    while (tokeniser.nextToken()) {
      tokenType = tokeniser.getTokenType();
      tokenValue = tokeniser.getStringValue();
      if (tokenType == PRTokeniser.TokenType.NUMBER) {
        buf.add(tokenValue);
      } else if (tokenType == PRTokeniser.TokenType.OTHER
                  & tokenValue.equals("m")
                  & startOfShape == 0) {
        highlighted.add(buf.get(buf.size() - 2));
        highlighted.add(buf.get(buf.size() - 1));
        startOfShape = 1;
      } else if (tokenType == PRTokeniser.TokenType.OTHER
                  & tokenValue.equals("S")
                  & startOfShape == 1) {
        highlighted.add(buf.get(buf.size() - 2));
        highlighted.add("" + pageNumber);
        startOfShape = 0;
      }
    }
  }
```

Listing 1. Searching for appropriate tokens

If a pattern is found (1), an index is created which stores information about the location of selection (coordinates with page number) (2):

$$\mathrm{x_1, y_1, x_2, pageNumber, \ldots, x_n, y_n, x_{n+1}, pageNumber} \qquad (2)$$

where:

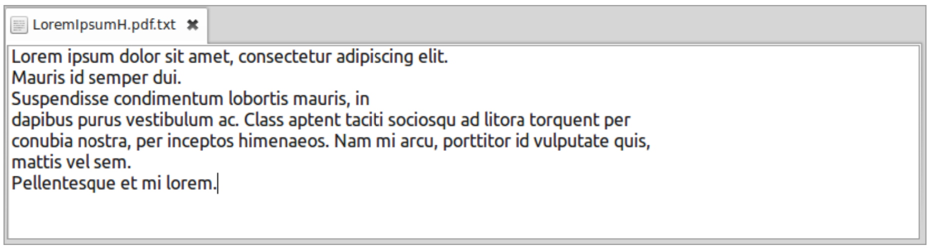
x_1, - x-coordinate of the beginning of the selection
y_1, - y-coordinate of the beginning of the selection
x_2, - x-coordinate of the end of the selection
pageNumber, - number of page in the document where the selection is placed

After searching the entire document, the method returns a list of the form (2), which is used to build the filter. The filter is a rectangle (position) collected sequentially from the index (2). Then the data are retrieved from the separated area and resulting file is created that contains content originally selected in the PDF document. The method used is shown at Listing 2

```
for (int i = 0; i < highlighted.size(); i+=4) {
  int x1 = (int)Double.parseDouble(highlighted.get(i));

  int y = (int)Double.parseDouble(highlighted.get(i+1));
  int x2 = (int)Double.parseDouble(highlighted.get(i+2));
  int pageNumber =
          (int)Integer.parseInt (highlighted.get(i+3));
  int fontSize = 12;
  Rectangle size = reader.getPageSize(pageNumber);
  Rectangle rect = new Rectangle(
                      x1, size.getTop()-y-fontSize,
                      x2, size.getTop() - y);

  RenderFilter filter = new RegionTextRenderFilter(rect);
  TextExtractionStrategy strategy;
  strategy = new FilteredTextRenderListener(
                new LocationTextExtractionStrategy(),
                filter);

  out.println(PdfTextExtractor.getTextFromPage(
                      reader, pageNumber, strategy));
}
```

Listing 2. Extracting info from selected areas

Below there is an attached screenshot showing the output file (result) of the program on the document with the selected text (Figure 2).



```
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Mauris id semper dui.
Suspendisse condimentum lobortis mauris, in
dapibus purus vestibulum ac. Class aptent taciti sociosqu ad litora torquent per
conubia nostra, per inceptos himenaeos. Nam mi arcu, porttitor id vulputate quis,
mattis vel sem.
Pellentesque et mi lorem.
```

Figure 2. Text document generated from selection

5 Summary

The goal of the paper was to provide a reader with an insight into the essentials of the REQ library, into the directions in which it is going to be enhanced and presented a very interesting solution to a problem of “impedance mismatch” between the compilation units and electronic documents, that are the major carriers of requirements information. The method covers the problem, yet evaluating it's effectiveness, from the point of view of a programmer or requirements engineer should (and will be) be a subject of some further research.

References

1. Davis A. M., Leffingwell D. A., 1995, *Using Requirements Management to Delivery of Higher Quality Applications*, Rational Software Corporation
2. Dardenne A., van Lamsweerde A., Fickas S., 1993, *Goal-directed Requirements Acquisition*, Science of Computer Programming, Vol. 20, pp. 3-50
3. Grzanek K., *Source Code-Oriented Requirements Management*, Computer Methods in Practice, A. Cader., M. Jacymirski, K. Przybyszewski (eds.), Academic Publishing House EXIT, 2012, Warszawa, pp. 21-45
4. *The Clojure Language Website*, 2013, http://clojure.org
5. Eisenbarth T., Koschke R., Simon D., 2003, *Locating Features in Source Code*, IEEE Transactions on Software Engineering, pp. 210-224
6. Eaddy M., Aho A.V., Antoniol G., Gueheneuc Y.G., 2008, CERBERUS: *Tracing Requirements to Source Code Using Information Retrieval, Dynamic Analysis, and Program Analysis*, ICPC 2008. The 16th IEEE International Conference on Program Comprehension, pp. 53-62
7. *Git, Website* 2013, http://git-scm.com/
8. Oracle Technology Network, 2013, *The Java Language Specification*, http://docs.oracle.com/javase/specs/
9. *iText, Website* 2013, http://itextpdf.com/
10. Lowagie B., 2007, *iText in Action, Creating and Manipulating PDF*, Manning Publications Co., ISBN 1932394796
11. *Xournal, Website* 2013, http://xournal.sourceforge.net/

STORE REVISITED

Konrad Grzanek

IT Institute, Academy of Science, Łódź, Poland
kgrzanek@spoleczna.pl, kongra@gmail.com

Abstract

Building abstraction layers is the key do the creation of reliable, scalable and maintainable software. Large number of database models and implementations together with the requirements coming from agile and TDD methodologies make it even more tangible. The paper is an attempt to present features and abstraction layers of a transactional key → value persistent storage library in which the physical storage is fully transparent for a programmer and exchangeable on the run-time.

Key words: Databases, transactions, functional programming, Clojure, software architecture

1 Introduction

The database research and development is one of the key regions of both scientific and industrial activities. The advent of no-SQL databases, massive data storage and processing environments like Google BigTable, Hadoop as well as many other large and small database and storage solutions make it a great ecosystem to design and build large-scale data-processing software, but sometimes makes the design decisions harder than expected. It is not easy to choose a concrete storage solution when both the functional and non-functional requirements are hard to freeze at the initial phases of software design and development. Moreover the programmers would be so much satisfied being able to write business logic abstracting away from the details of a concrete storage. It is also very important in TDD and other agile methods.

The paper summarizes new features and some implementation details of a flexible ***Store*** library, a transactional and realization-transparent key → value storage abstraction being an enhanced version of a previously implemented solution [1]. It also gives an insight into ways of using the library and implementing new realizations.

2 Layers of Abstraction

Previous version of the ***Store*** library as described in [1] was designed and implemented with the following assumptions:

- Key → value store being the major data storage model. Effectively this meant abandoning the relational model for good.
- Query language is the application (business) logic language. Instead of SQL we use the operators and procedures of the high level programming language. This decision might appear costly with respect to joins, but the key → value store architecture does not support them anyway.
- ***A lack of transactions.***
- ***Only a single physical storage implementation using Berkeley DB Java Edition*** [3].

A diagram depicting the architecture of the original solution can be found in [1]. When approaching a ***Store*** renewal we decided to make the following improvements in various aspects of the whole:

- **Add the transaction processing**, making it an option if possible 1.
- Make no assumptions about the physical storage. There should be **various and pluggable storage realizations**.
- It should be possible to change the storage realizations on the run-time without stopping the application.
- All the key elements of the system (also the realizations) should be as loosely coupled as possible, reconfigurable on the run-time and provably safely managed (without any resource leaks).
- All the key elements of the system should exhibit solely (provably) correct multitasking behaviors.

The following Figure 1 presents the architecture of the new ***Store***[2].

1 In many transactions-supporting systems it is impossible to turn the transactions off. Instead an auto-commit mode may be used, like in the case of JDBC. Berkeley DB JE environments support turning the transactions on and off on the environment opening. When opened in a transactional mode, it also supports auto-commit [4].

2 Including Repo – a high level objects storage library. For more information see [2].

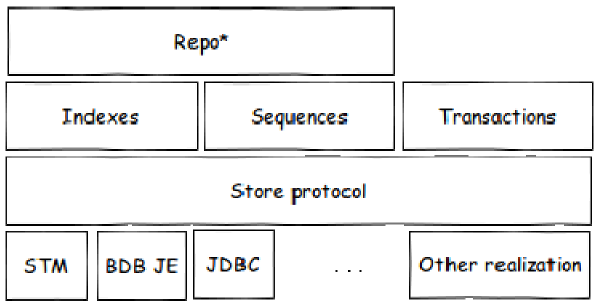


Figure 1. The architecture of the Store library

In the following sections all the elements of the architecture will be described with particular emphasis on the abstractions, but also with some implementation details. From now on we assume the following compilation unit header:

```
(ns <ns-name>
  (:use     [kongra.core])
  (:require [kongra.store :as S]))
```

3 Sequences

The nature of all objects representing the basic abstractions in ***Store*** was intended to be lightweight, i.e. their creation was supposed to be cheap in terms of CPU and memory usage. Being “lightweight” also means accessing their realization (possibly persistent, resource-heavy) is delayed for as long as possible and, the created abstraction does not hold any details of the realization within itself. This is essential with respect to the planned substitutions of realizations on the run-time.

An abstraction of a persistent, transactional sequence may be created using the following expression:

```
(S/sequence "<name>")
```

The created abstraction represents a sequence of step (delta) 1 and the initial value 1. To modify the settings another form may be used:

```
(S/sequence "<name>" <start> <step>)
```

where the parameters *start* and *step* are Clojure long integral values. The creation is pretty cheap indeed:

```
(microbench-repeat* 10 1e6 5 (S/sequence "<name>"))

Warming up for 5 [s] ...
Benchmarking...
Total runtime:  151.92165799999998 msecs
Highest time :  40.679912  msecs
Lowest time  :  8.573213  msecs
Average      :  12.833566624999996  msecs
```

That simple benchmark performed on a low-commodity machine[3] shows that the creation of 1 mln sequence handles takes circa 13 milliseconds. One is encouraged to use the form to access sequence abstraction as needed. In particular it is more elegant to use the *(S/**sequence** ...)* form in other S-expressions than to:

```
(def s (S/sequence …)
```

and then use the ***s*** symbol.

The following table shows all the sequence operators in ***Store*** with their semantics:

Table 1. Sequence operators and their semantics

Operator	Argument(s)	Meaning
next!	[seq]	Generates a next sequence value.
recent	[seq]	Returns the recently generated value.
drop!	[seq]	Deletes (drops) the passed sequence. This operation causes the underlying realization free all resources related with this sequence abstraction. The sequence abstraction is not deleted and may be used in further operations.

An example interactive sequence session (in Clojure REPL within a namespace ***stest***) can be seen below:

```
stest> (S/next!  (S/sequence "1st-seq" 0 1))
0
stest> (S/next!  (S/sequence "1st-seq" 0 1))
```

3 AMD E-450 netbook running in the *performance-on-demand* mode, 4GB of RAM, JVM settings: -Xms128m -Xmx2g -XX:MaxPermSize=256M -XX:+UseCompressedOops -XX:+UseParallelGC

```
1
stest> (S/next!  (S/sequence "1st-seq" 0 1))
2
stest> (S/recent (S/sequence "1st-seq" 0 1))
2
stest> (S/drop!  (S/sequence "1st-seq" 0 1))
OperationStatus.SUCCESS
stest> (S/next!  (S/sequence "1st-seq" 0 1))
0
stest> (S/next!  (S/sequence "1st-seq" 0 1))
1
stest> (S/next!  (S/sequence "1st-seq" 0 1))
2
```

Few things are worth mentioning here:

– There is no need to perform any explicit initialization of a sequence abstraction nor it's realization.
– The actual realization is fully transparent. The only realization-dependent value above is `OperationStatus.SUCCESS` – a result of an underlying operation of dropping the sequence in a Berkeley DB JE Store being the currently used *store*4 realization.
– No special steps have be taken between dropping the sequence and reusing it. This is a manifestation of the already mentioned overall lightweight nature of the **Store** abstractions.

Dropping a sequence that has never been referenced before also exhibits a desired behavior:

```
stest> (S/drop! (S/sequence "2nd-seq"))
OperationStatus.NOTFOUND
```

It is so in the case of a Berkeley DB JE realization and should be in all other realizations.

4 Indexes

An index is a key → value storage (mapping) abstraction. To get an access to an index one has to use an expression:

4 We use a capitalized name ***Store*** to refer to a library, and a lowe-case name *store* when talking about the possible realization(s) of Store abstractions, in fact – realizations of the kongra.store.Store *protocol* as will be described further.

```
(S/index "<name>")
```

or

```
(S/index "<name>" <key-type> <value-type>)
```

The second version evaluates to an index abstraction. Informing about *key-type* and *value-type* may be necessary in some realizations, when the serializations of key and values is non-trivial – JDBC or Berkeley DB JE realizations are the apparent examples here. The values of both *key-type* and *value-type* may be any Clojure or Java objects/values, as long as the realizations that expect them are able to recognize their meaning[5]. We strongly recommend using Java classes, Clojure *keywords* or *symbols*.

Like in the sequences case the actual realization of the index depends on the currently present *store*. There are the following index operations:

Table 2. Index operators and their semantics

Operator	Argument(s)	Meaning
get	[index key]	Returns a value for a given key stored in the index or nil, when no value present.
put!	[index key val] [index & kvs]	Puts the key → val entry into the index. Allows passing a sequence of key → val pairs to put them all in a single call (see example below).
del!	[index key] [index & ks]	Deletes a entry for the key. Allows passing a sequence of keys to delete them all in a single call.
drop!	[index]	Deletes (drops) the passed index. This operation causes the underlying realization free all resources related with this index abstraction. The index abstraction is not deleted and may be used in further operations.
clear!	[index]	Deletes all entries from the index.
entries	[index]	Returns a collection of [key value] pairs representing all entries of the index. The laziness of the returned collection depends on the *store* realization.

5 How to inform the realizations of the *key*- and *value-type* semantics is a question of the realizations, not the Store abstraction and lays beyond the scope of this paper.

To become familiar with these operations, please take a look at the following example:

```
stest> (def persons (S/index "persons" :long String))
#'stest/persons
stest> (S/get persons 77032403124)
nil
stest> (S/put! persons 77032403124 "Jan Kowalski")
OperationStatus.SUCCESS
stest> (S/get persons 77032403124)
"Jan Kowalski"
stest> (S/put! persons 77032403124 "Jan Kowalski"
                       77032403125 "Anna Kowalska")
nil
stest> (S/get persons 77032403125)
"Anna Kowalska"
stest> (S/put! persons 1 "Roman" 2 "Piotr" 3 "Jerzy")
nil
stest> (S/get persons 1)
"Roman"
stest> (S/get persons 2)
"Piotr"
stest> (S/get persons 3)
"Jerzy"
stest> (S/del! persons 1 2)
nil
stest> (S/get persons 1)
nil
stest> (S/get persons 2)
nil
stest> (S/get persons 3)
"Jerzy"
stest> (doclean (doall (S/entries persons)))
((3 . Jerzy))
```

The last expression (S/entries persons) returns a lazy sequence of entries stored in a Berkeley DB JE *store* in this case. But the iteration over the persistent entries requires opening a related *Database* cursor and so a *doclean* context is required to clean up all interconnected resources. In the case of other store realizations (e.g. Software Transactional Memory) this may not be necessary.

5 Transactions

Adding the transactions support was one of the two most important goals leading to the ***Store*** redesign and re-implementation. Table 3 summarizes the new transactional abstraction.

Table 3. Transactional operators and their meaning

Operator	Argument(s)	Meaning
`within-transaction`[6]	`[& body]`	Executes the body of code within a transaction, if there is a transaction running in the current scope. When no transaction present, runs within a new transaction.
`within-new-transaction`[6]	`[& body]`	Executes the body of code within a newly created transaction. Commits when the body executes without any errors/exceptions and rolls-back the transaction when there were errors/exceptions propagated.
`asserting-transaction`[6]	`[& body]`	Asserts the presence of a transaction and then executes the body. Raises an error when no transaction present.
`commit`	`[]`	Commits the currently running transaction. This operation should only be performed when absolutely necessary. A preferred way to go is auto-committing ***within-transaction*** or ***within-new-transaction***.
`rollback`	`[]` `[save-point]`	Aborts (rolls-back) the currently running transaction. As in the case of ***commit*** this operation should also be used only when absolutely necessary, e.g. when rolling back to a previously created save-point[7].

6 Implemented as Clojure macro [8]

7 Whether or not save-points functionality will be supported depends on a concrete underlying *store* realization.

set-savepoint	[]	Creates (sets) a new save-point[7].
release-savepoint	[save-point]	Releases (frees) a save-point[7].

Note that it is possible to use (***within-new-transaction*** …) inside a scope of an already running transaction. Syntactically this looks like nesting transactions however the actual support for nested transactions depends on a concrete underlying store realizations. Currently neither the Clojure STM nor the Berkeley DB JE realizations do not support this feature. One can expect a support for nested transactions from the JDBC realizations, as some RDBMs provide this feature[8] (like Oracle).

Anyway, the following good practices should be applied when using ***Store*** to implement a transactional system:

– By default use (***within-transaction …)*** to execute a body of code in a transactional context ensuring a presence of this context.
– Use (***asserting-transaction …)*** to execute a body of code in a transactional context, when the lack of a context is perceived as an erroneous state.
– Do not **commit** or **rollback** explicitly if possible
– Use save-points with care.
– Do not design with nested transactions8.

The following procedure written in a transactional way creates one million key → value pairs and puts them into an index.

```
(defn perftest
  []
  (let [s       (S/sequence "persons-seq" 0 1)
        persons (S/index "persons" Long String)]
    (S/within-transaction
     (dotimes [i 1e6]
       (S/put! persons (S/next! s)
                       (str "John Doe-" i))))))
```

achieving the following performance[9]:

```
stest> (time (perftest))
"Elapsed time: 32282.751323 msecs"
nil
```

8 Please, note that nesting transactions is perceived by some software and database engineers as a symptom of badly designed application.

9 The experiment was run on a previously specified machine with a Berkeley DB JE *store* realization opened with the following settings: ***transactional***, ***cache-size*** 512MB, ***cache-mode*** DEFAULT**,** ***durability*** COMMIT_SYNC, ***lock-mode*** READ_COMMITTED (see [3], [4]).

The particular *store* realization[9] used to perform the simple benchmark above depends heavily on using the database cache:

```
stest> (room)
Used memory  : 580,949692 [MB]
Free memory  : 455,550308 [MB]
Total memory : 1036,500000 [MB]
Max memory   : 1820,500000 [MB]
nil
stest> (gc)
Used memory  : 139,630356 [MB]
Free memory  : 767,119644 [MB]
Total memory : 906,750000 [MB]
Max memory   : 1820,500000 [MB]
nil
```

The significant JVM heap usage after a garbage-collection (circa 140 MB) is a manifestation of a way the Berkeley DB JE manages, and – in fact – takes an advantage of using the cache.

Finally one final note on the *auto-commit* mode. Store realizations should support this mode. Only when it is impossible to be implemented, it may be permitted to skip this feature. Such a situation probably will never occur, as most transactional database or library providers offer some kind of an auto-commit. On the other hand, developers using the Store should be aware of the fact that running large amounts of operations in an auto-commit mode may decrease the overall systems' performance beyond the point of their usability.

6 The Store Abstraction

After presenting the high level abstractions of the ***Store*** library, it is time now to take a look at the core abstraction, namely the ***kongra.store.Store*** *protocol* (protocols as means of expression in Clojure are described in depth in [9]). The protocol definition looks as follows:

```
(defprotocol Store
  (sequence-drop!          [this seq])
  (sequence-next!          [this seq])
  (sequence-recent         [this seq])

  (index-drop!             [this index])
  (index-get               [this index key])
  (index-put!              [this index key val]
                           [this index key val kvs])
  (index-del!              [this index key]
                           [this index key ks])
  (index-clear!             [this index])
  (index-entries            [this index])

  (tx-within-transaction     [this body])
  (tx-within-new-transaction [this body])
  (tx-asserting-transaction  [this body])
  (tx-commit                 [this])
  (tx-rollback               [this] [this savepoint])
  (tx-set-savepoint          [this])
  (tx-release-savepoint      [this savepoint]))
```

To provide a *store* (*kongra.store.Store* protocol) realization a provider has to do the following:

– introduce a new type whose object (or objects) would represent the realization,
– implement all the protocol methods.

A part of an example realization (with Berkeley DB JE) looks like below:

```
(defrecord ^:private JEStore [env])
(def INSTANCE (JEStore. (JE/env)))

(extend-protocol S/Store
  JEStore

  (sequence-drop!  [this s] …)
  (sequence-next!  [this s] …)
  (sequence-recent [this s] …)

  (index-drop! [this index]     …)
  (index-get   [this index key] …)

  (tx-within-transaction [this body] …)
  (tx-within-new-transaction [this body] …)

  (tx-asserting-transaction [this body]…)

  (tx-commit [this] …)

  (tx-rollback
    ([this]            …)
    ([this savepoint]  …))

  (tx-set-savepoint [this] …)

  (tx-release-savepoint [this savepoint] …))
```

The Berkeley DB JE *store* realization is the default one. Getting the current *store* realization is possible with a `(S/store)` call. As it was mentioned earlier, there are two ways to change the realization during the run-time. One is calling `(S/set-store! <store>)`. This causes a persistent, all-threads change of the default *store* realization. Another way is to use a dynamic binding of a store to a new one and executing a body of code within this newly established binding, e.g.:

```
(time (S/with-store STM/INSTANCE
                  (perftest)))
"Elapsed time: 8603.678825 msecs"
nil
```

Now, the *perftest* was executed with a Software Transactional Memory realization represented by ***STM/INSTANCE*** value. The operation results in much lower execution time (even considering the influence of database caching in the previous example) and larger memory consumption:

```
stest> (gc)
Used memory  : 297,057205 [MB]
Free memory  : 630,130295 [MB]
Total memory : 927,187500 [MB]
Max memory   : 1820,500000 [MB]
nil
```

Certainly the biggest win with this approach is the ability to change the target *store* realizations algorithmically on the run-time. It is the other most significant benefit coming from the ***Store*** redesign and re-implementation.

7 Parallelism of Abstractions and Realizations' Details

The most important problem arising within the ***Store*** library and the *store* realizations as seen from the point of view of potential users (programmers and system designers) is the complexity of configuration behind the particular realizations. With a naive approach we would be tempted to put as much configuration details into the *kongra.store.Store* protocol, in fact – trying to find a kind of a “greatest common denominator” for all possible configuration options. Even in the first look this seems a bad design. The new ***Store's*** implementer decided to go a completely different way, eliminating all configuration out of the ***Store*** abstractions and leaving them in current and future realizations. This convenient and flexible approach gives a rise of a problem of a loss of the abstract nature of source codes written using ***Store***. Codes using the library should abstract away from the details of the realizations. One (and currently preferred) approach is to put all realization-dependent codes in a place dedicated to build a kind of a configuration context and the abstract business logic codes in all other places (procedures, modules) and then to connect them transparently using the dynamic variables and in-thread bindings. To get a sample how this can be done, please take look at the following piece of code:

```
(BDBJE/with-lock-mode BDBJE/LOCK-MODE-READ-UNCOMMITTED
    (BDBJE/with-lock-timeout 100 ;; msecs
      (perftest)))
```

where the locking mode present in the Berkeley DB JE wrapper library for Clojure (not even the *store* realization) is defined like below:

```
(def LOCK-MODE-DEFAULT          LockMode/DEFAULT)
(def LOCK-MODE-READ-COMMITTED   LockMode/READ_COMMITTED)
(def LOCK-MODE-READ-UNCOMMITTED LockMode/READ_UNCOMMITTED)
(def LOCK-MODE-RMW              LockMode/RMW)

(dyndef *lock-mode* nil)

(defn lock-mode
  []
  (dynval *lock-mode*))

(defn set-lock-mode!
  [lmode]
  (dynset! *lock-mode* lmode))

(defmacro with-lock-mode
  [lmode & body]
  `(binding [*lock-mode* ~lmode] ~@body))
```

8 Clojure STM Realization

The Software Transactional Memory is a modern approach to mutual-exclusion problems in multitasking environments. Clojure provides it's realization with references, atoms, agents, dynamic variables and persistent data structures [8], [9]. We decided to use STM for a *store* realization, even though STM lacks durability feature of a full ACID transactional system (all data is stored in RAM). It's unquestionable advantage is speed.

The major STM *store* realization features are:

– A *sequence* is a name → value pair within a mapping reference.
– An *index* is a mapping reference.

- No support for nested transactions, as the STM does not support them10.
- No support for save-points[10].
- No support for an explicit ***commits***[10].
- The explicit ***rollbacks*** are possible but not to the save-points10, only parameter-less.
- Data do not undergo any conversions on the read/write. The types declared on referring to an *index* are not used. This increases the operation speed, but eliminates type-safety by disabling any type-checks.

9 Berkeley DB JE Realization

The Berkeley DB Java Edition library [3], [4] offers full ACID stack, it also allows to operate in a non-transactional mode when properly configured on the opening-time. Our BDB JE *store* realization has the following properties:

- No support for save-points10, as in the case of the STM realization.
- Support for explicit commits and rollbacks, but not to the save-points[10].
- A support for nested transactions in the *store* realization layer. However the Berkeley DB JE library does not support this feature by now. Whether or not it will be supported is a question of the future.

10 A Discussion on Possible Future Realizations

The most promising, yet not existing realization of the ***Store*** library abstractions is the one using JDBC and relational databases. There are many ways a key → value storage may be implemented within a relational model. Readers are encouraged to take part in a research on the most effective ways of implementing this realization and also in design and programming phase.

Another interesting way to go is to use the Hadoop distributed data storage and processing engine. This direction is particularly important, as the massive storage and distributed data processing seems to be the central point in current and future database research and production use.

References

1. Grzanek K., 2010, *Store: Embedded Persistent Storage For Clojure Programming Language*, Journal of Applied Computer Science Methods No. 1 Vol. 2 2010, pp. 83

10 A Log4J warning is emitted on an attempt to use the feature.

2. Grzanek K., 2011, *Repo: High-Level Persistence Layer for Clojure*, Journal of Applied Computer Science Methods No. 2 Vol. 2 2011, pp. 5-16
3. *Oracle Berkeley DB Java Edition*, 2013, Website: http://www.oracle.com/technetwork/database/berkeleydb/overview/index-093405.html
4. *Oracle Berkeley DB Java Edition, Getting Started with Transaction Processing*, 2008, Release 3.3
5. DeCandia G., Hastorun D., Jampani M., Kakulapati G., Lakshman A., Pilchin A., Sivasubramanian S., Vosshall P., Vogels W., 2007, *Dynamo: Amazon's Highly Available Key-value Store*, SOSP'07, October 14–17, 2007, Stevenson, Washington, USA, pp. 205-220
6. Schütt T., Schintke F., Reinefeld A., 2008, *Scalaris: reliable transactional p2p key/value store*, Proceedings of the 7th ACM SIGPLAN workshop on ERLANG, pp. 41-48
7. Lim H., Fan B., Andersen D., Kaminsky M., 2011, *SILT: a memory-efficient, high-performance key-value store*, Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 1-13
8. Halloway S., 2009: *Programming Clojure*, ISBN: 978-1-93435-633-3, The Pragmatic Bookshelf
9. Emerick Ch., Carper B., Grand Ch., 2012, *Clojure Programming*, O'Reilly Media Inc., ISBN: 978-1-449-39470-7

Information for Authors

Authors are invited to submit original papers, within the scope of the JACSM, with the understanding that their contents are unpublished and are not being actively under consideration for publication elsewhere.

For any previously published and copyrighted material, a special permission from the copyright owner is required. This concerns, for instance, figures or tables for which copyright exists. In such a case, it is necessary to mention by the author(s), in the paper, that this material is reprinted with the permission.

Manuscripts, in English, with an abstract and the key words, are to be submitted to the Editorial Office in electronic version (both: source and pdf formats) via e-mail. A decision to accept/revise/reject the manuscript will be sent to the Author along with the recommendations made by at least two referees. Suitability for publications will be assessed on the basis of the relevance of the paper contents, its originality, technical quality, accuracy and language correctness.

Upon acceptance, Authors will transfer copyright of the paper to the publisher, i.e. the Academy of Management, in Lodz, Poland, by sending the paper to the JACSM Editorial Office.

The papers accepted for publication in the JACSM must be typeset by their Authors, according to the requirements concerning final version of the manuscripts. However, minor corrections may have been carried out by the publisher, if necessary.

The printing area is 122 mm × 193 mm. The text should be justified to occupy the full line width, so that the right margin is not ragged, with words hyphenated as appropriate. Please fill pages so that the length of the text is no less than 180 mm.

The first page of each paper should contain its title, the name of the author(s) with the affiliation(s) and e-mail address(es), and then the abstract and keywords, as show above. Capital letters must be applied in the title of a paper, and 14 pt. boldface font should be used, as in the example at the first page. Use: 11-point type for the name(s) of the author(s), 10-point type for the address(es) and the title of the abstract, 9-points type for the abstract and the key words.

For the main text, please use 11-point type and single-line spacing. We recommend using Times New Roman (TNR) fonts in its normal format. Italic type may be used to emphasize words in running text. Bold type and underlining should be avoided. With these sizes, the interline distance should be set so that some 43 lines occur on a full-text page.

The papers should show no printed page numbers; these are allocated by the Editorial Office.

Authors have the right to post their Academy of Management-copyrighted material from JACSM on their own servers without permission, provided that the server displays a prominent notice alerting readers to their obligations with respect to copyrighted material. Posted work has to be the final version as printed, include the copyright notice and all necessary citation details (vol, pp, no, …).

An example of an acceptable notice is:
"This material is presented to ensure timely dissemination of scholarly work, and its personal or educational use is permitted. Copyright and all rights therein are retained by the Copyright holder. Reprinting or re-using it in any form requires permission of the Copyright holder.

The submission format for the final version of the manuscript
and the MS Word template are available on our web site at:
http://www.acsm.swspiz.pl

Journal of Applied Computer Science Methods

Contents