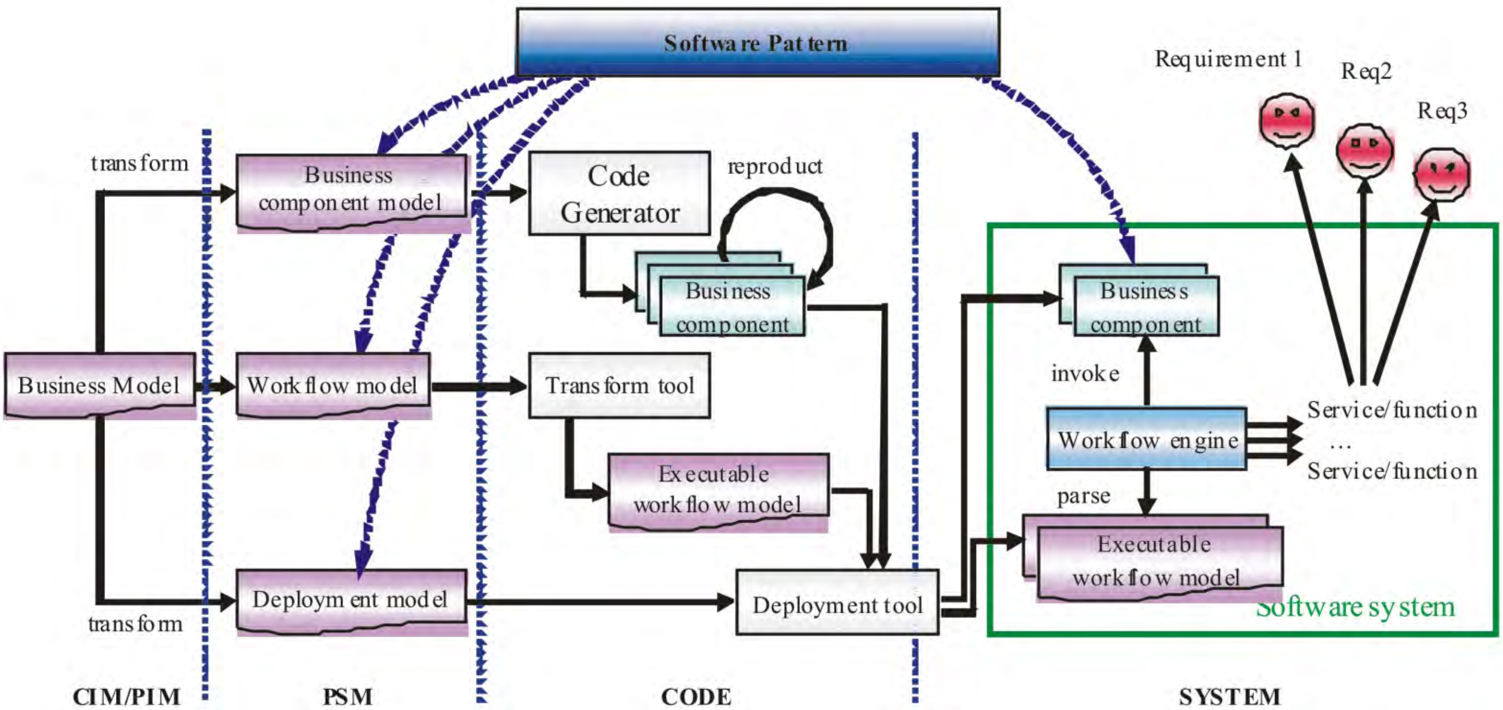
ISSN: 1945-3116 Vol. 3, No. 10, October 2010

Scientific Research

Journal of Software Engineering and Applications



ISSN: 1945-3116

www.scirp.org/journal/jsea

Journal Editorial Board

J. Software Engineering & Applications, **2010, 3, 901-1004**
Published Online October 2010 in SciRes (http://www.SciRP.org/journal/jsea/)

TABLE OF CONTENTS

Volume 3 Number 10 October 2010

Journal of Software Engineering and Applications (JSEA)

Journal Information

SUBSCRIPTIONS

The *Journal of Software Engineering and Applications* (Online at Scientific Research Publishing, www.SciRP.org) is published monthly by Scientific Research Publishing, Inc., USA.

Subscription rates:

Print: $50 per issue.

To subscribe, please contact Journals Subscriptions Department, E-mail: sub@scirp.org

SERVICES

Advertisements

Advertisement Sales Department, E-mail: service@scirp.org

Reprints (minimum quantity 100 copies)

Reprints Co-ordinator, Scientific Research Publishing, Inc., USA.

E-mail: sub@scirp.org

COPYRIGHT

PRODUCTION INFORMATION

For manuscripts that have been accepted for publication, please contact:

E-mail: jsea@scirp.org

J. Software Engineering & Applications, 2010, 3, 901-905
doi:10.4236/jsea.2010.310106 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

AS-Level Internet Macroscopic Topology Centralization Evolvement Analysis

Jie Song, Hai Zhao, Bo Yang, Huali Sun

The School of Information Science & Engineering, Northeastern University, Shenyang, China.
Email: moyuan87@163.com

Received July 1st, 2010; revised August 1st, 2010; accepted August 5th, 2010.

ABSTRACT

The As-level topology is a hotspot of the recent reseaches. We can understand the centralization of the network clearly by researching the evolvement trend of the Internet macroscopic topology. The massive data we use in this paper is from CAIDA (The Cooperative Association for Internet Data Analysis) Skitter project. And the time span of the data is from July, 2001 *to January,* 2008. *This paper introduces the background of the AS-level topology at first, then carries out the evolvement of degree, core and layer. It is believed that the influence of the top-degree nodes on the other nodes decreases and the centralization of network is going to fall off with the decrease of the core. And the nucleus status of network declines.*

Keywords**:** *AS-Level, Centralization, Top-Degree, Core, Layer*

1. Introduction

The Internet has been a complex self-organizing ecosystem which is composed of numerous computers and is now quite different from its prototype, ARPANET. Although the Internet was built up by human personally, none of us can describe what it looks like and how it operates. The research on the Internet topology is to study the laws which are contained in the network which seems chaotic. It is inherently necessary to know the Internet by recognizing the inner mechanism of the network topology, the basis for developing and utilizing Internet in a higher level.

Since the Internet was born, the studies on it have been endless. In the early years people were more concerned about the researches on the Internet architecture, the network protocols, the connections between the computers, as well as the service on the Internet. The results of the study on the complex science and complex networks made the researchers at home and abroad come to realize that the Internet has been one part of the complex networks in recent decades. Consequently, people began to research the Internet from the angle of the complexity and complex networks [1]. Recent years, people have made a remarkable progress in this area and have found a great number of characteristic laws which are hidden in the network. However problems still remain such as the small space, the short span of the data analysis and the simple method in measuring. So we have to do more comprehensive and further studies on it [2].

The researches on the Internet topology mainly concentrates on AS-level (Autonomous System) and routing-level. AS-level topology is in a higher level compared to the routing-level topology in the network [3]. Its characteristics and changes have a great influence on the Internet. Meanwhile, since the scale of data of the AS-level topology is small, we can implement the computation and analysis more effectively with deep-seated, high time complexity and explore the objective characters in more depth using the existed computer power [4]. More systematic analysis strategies and reliable analysis measures based on the results concluded by analyzing the AS-level topology can be summarized to study the laws in the router-level topology and to provide available means to statistical analysis for network with ultra large scale. Based on the main research trend in the recent years, we take the application value and the computational cost in the practical analysis into account. So we will focus on the data of the AS-level Internet topology.
Considering that the Internet is dynamic and its topology changes with time, we can further understand the centralization of the network by studying the trend of the inner topology evolvement.

2. Evolvement Analysis of Degree

Archipelago (Ark) is the newest active measurement infrastructure, the next generation in evolution of the Skitter infrastructure which CAIDA has operated for nearly a decade [5]. But in this paper we will still use the measuring data of the CAIDA Skitter because the time of the appearance of the Ark is short. The measurement of the global Internet data provided by the CAIDA Skitter meets the results of the statistical analysis data. And the results concluded by statistically analyzing should be much better to reflect the status of the Internet. So all the calculations involved in this paper use the data provided by the CAIDA Skitter. The time span is from July, 2001 to January, 2008.

The number of nodes is a variable to measure the scale of the Internet. The **Figure 1** shows us the evolvement trend of the number of nodes. X-axis is the time span of the actually measuring data from July, 2001 to January, 2008. Y-axis is the number of nodes. We can conclude that the number of nodes increased violently and then declined gradually in the first three years. The number in the last year declined stably with time. But the whole trend in this Fig increased in this period. What this Fig tells us is that the Internet is instable in the first three years but the performance of the Internet gradually turns better and more stable than before. In short, the scale of the Internet is larger than before.

The node degree is defined as the number of the links indicating to this node. It is one of the most used measurements in the topological analysis. It shows how hot these nodes are. If the degree of one node is large, it will

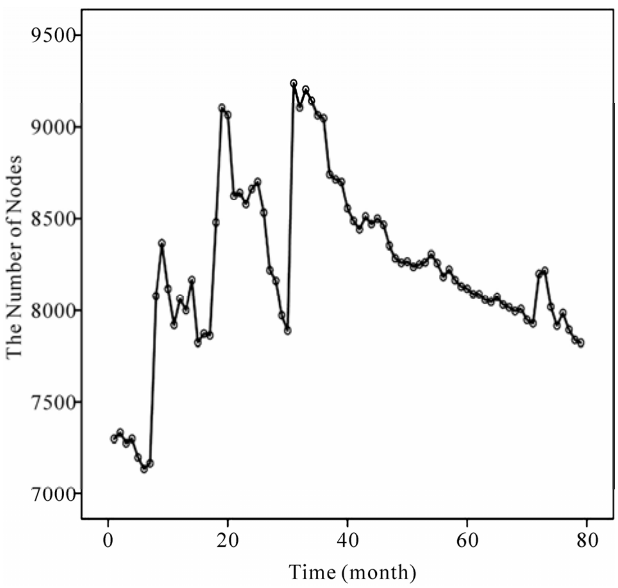


Figure 1. The number of nodes statistically changes with time.

explain that this AS connects massive other AS and has a heavy task in the Internet.

The average degree of network is a basic characteristics of network topology. The larger it is, the more links the network has. At the same time, the network is likely to have a better robustness [6].

From the **Figure 2(a)** we can conclude the law that the average degree of AS-level Internet varies from time to time. X-axis is the time span of the actually measuring data from July, 2001 to January, 2008. Y-axis is the average degree of network. The value changed between 5 and 6.8 with small amplitude but decreased in the whole time with the increase of the scale of the Internet. That is to say that the connectivity of the whole network decreases in this period.

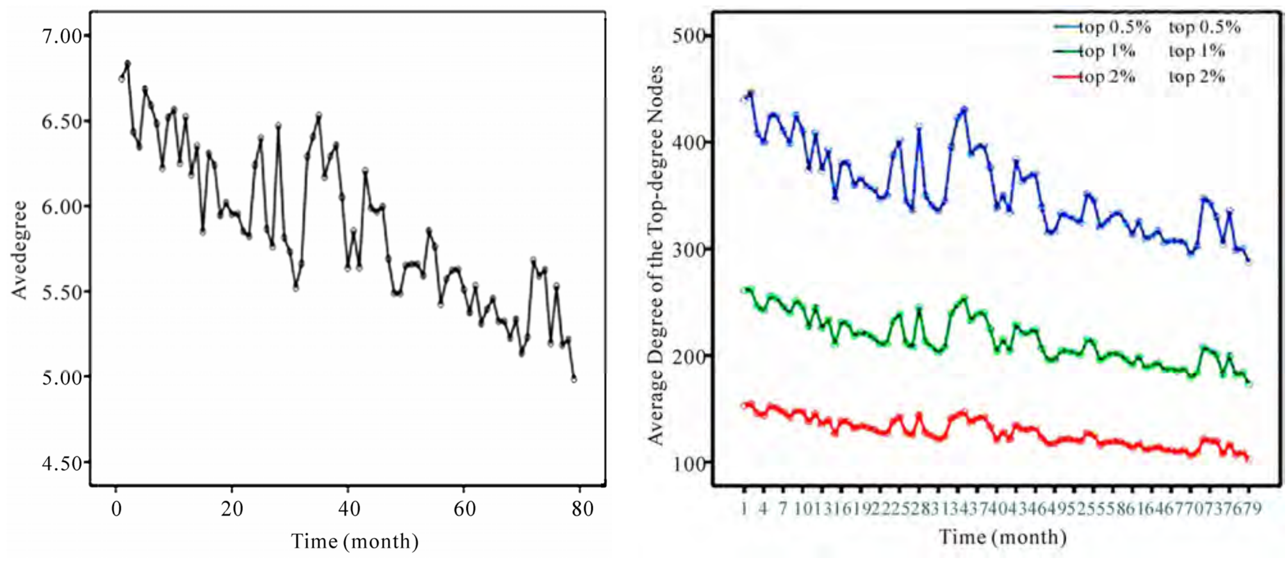
Figure 2(b) points out the average degree of the 0.5%, 1% and 2% of the top-degree nodes in the AS network evolves over time. X-axis is the time span of the actually measuring data. Y-axis is the average degree of these three sets. The value decreases slowly with time. The degree of the top-degree nodes declines with the drop of network average degree. The shapes of these curves in the **Figure 2(b)** are close and are similar to that in the **Figure 2(a)**. It indicates that the decline of the degree of the top-degree nodes leads to the drop of network average degree.

There are some exceptional AS in the data that we have got from the CAIDA. The degree of these nodes usually is small but may change greatly in some months. However, these exceptional actions would not last for along time. For example, there is an AS whose degree usually changes between 30 and 60 but rises to 4648 and 4596 in April and May of 2004 suddenly. Then it returns to normal level after July of 2004. We delete these exceptional situations in this paper in order to make the analysis reasonable.

From **Figure 3** we can conclude that the maximum node degree in AS-level Internet topology changes with time. X-axis is to show a time span of the actually measuring data. Y-axis is to show the maximum degree. The value in this Fig shocks heavily with the whole trend declining. That is to say the impact of the "hot" nodes which are the top-degree nodes gradually drops over time.

3. Evolvement Analysis of Core and Layer

From a given graph $G=(V,E)$ [7], we recursively delete all nodes whose degree are less than k and lines incident with these nodes. These nodes and links we have removed make up the set W which is called $k-1$ layer. And the remained sub-graph is called k-core [8]. The value of k decides the position of node in the topology graph. The highest one is called the core of this topology.



(a) (b)

Figure 2. The average degree statistically changes with time. (a) The average degree of network statistically changes with time; (b) The average degree of the top-degree nodes statistically changes with time.

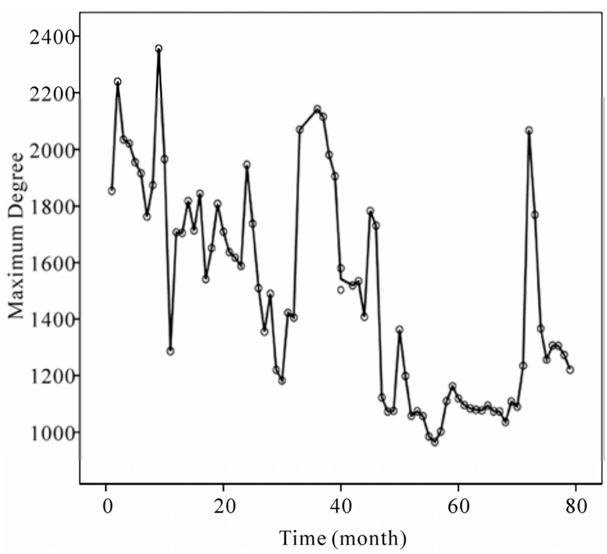


Figure 3. The maximum degree of network statistically changes with time.

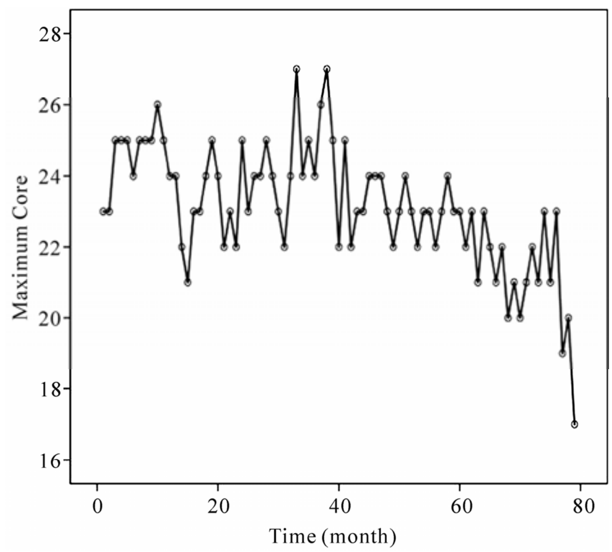


Figure 4. Core of the Internet statistically changes with time.

As one of important features of the Internet topology, core is a more complex measurement than the node degree to measure the connectivity of the nodes. The core of the nodes sometimes reflects the position of the node in the topology. Those with a high-core have more links and are more important to the Internet. They are in the inner layer of the Internet. We can see the hierarchical structure of the Internet from the nut to the outer space by analyzing the topology core

We can see from the definition of the core that the core of the graph is decided by the highest core. The lost of links of the nodes with high-core may lead to the remove of these nodes. Doing this may lead the same bad impact on the neighboring nodes and the core of the whole topology declines. X-axis is the time span of the actually measuring data. Y-axis is the network core. The value changes with small amplitude but declines in the last several years. The layer of the Internet is going to reduce.

The **Figure 5** shows us that the number of nodes declines fast in the low-layer but hardly changes in the high-layer. X-axis is the layer of the Internet in July each year. Y-axis is the node number of each layer. The **Figure**

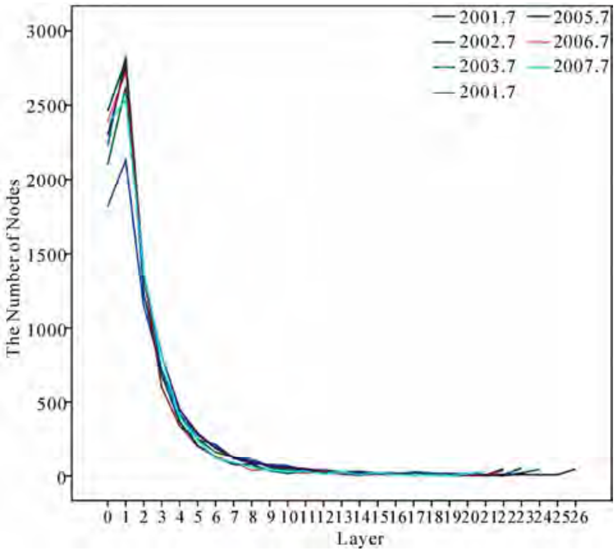


Figure 5. The number of nodes statistically changes with layer.

5 shows the situation: a small amount of the nodes in the high-layer but a greater number of nodes in the low-layer. The nodes in the high-layer are the nucleus of the Internet topology.

The **Figure 6** shows that the number of links gradually declines with the increase of the layer but increases suddenly in the high-layer. The decline tendency of the number is slow in the middle-layer. X-axis is to show the layer of network in July each year from 2001 to 2007. Y-axis is to show the number of links. When the layer of the Internet is high enough, the number of links would not decline but with some amplitude. The top-degree nodes are usually in a high-layer. So the number of links may change obviously once the number of these nodes

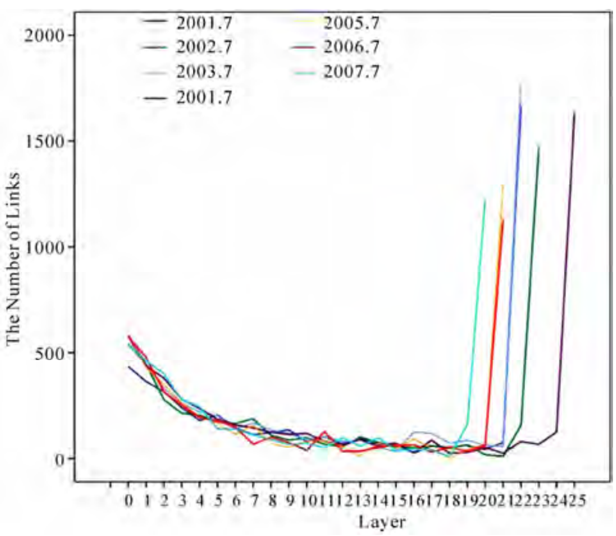
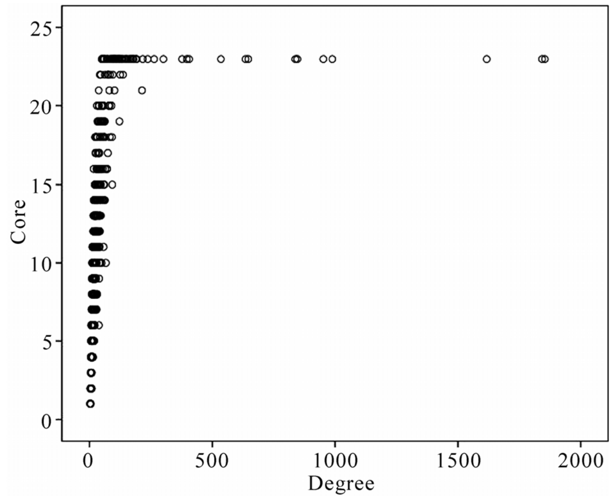
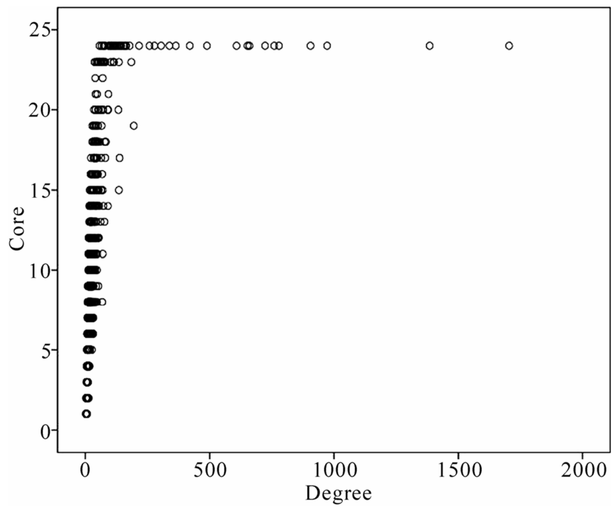


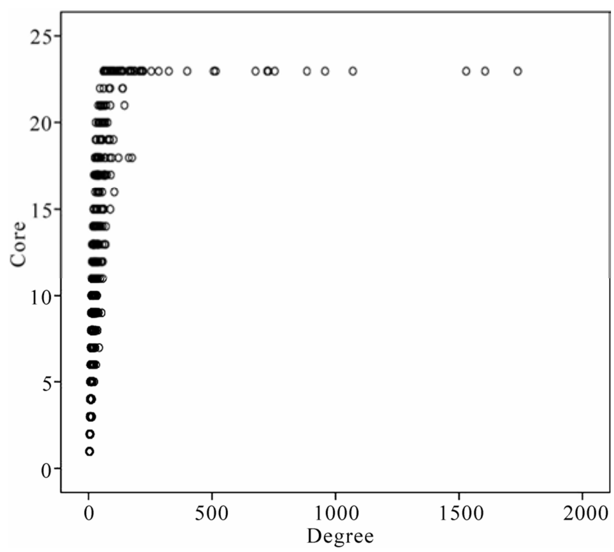
Figure 6. The number of links in each layer.



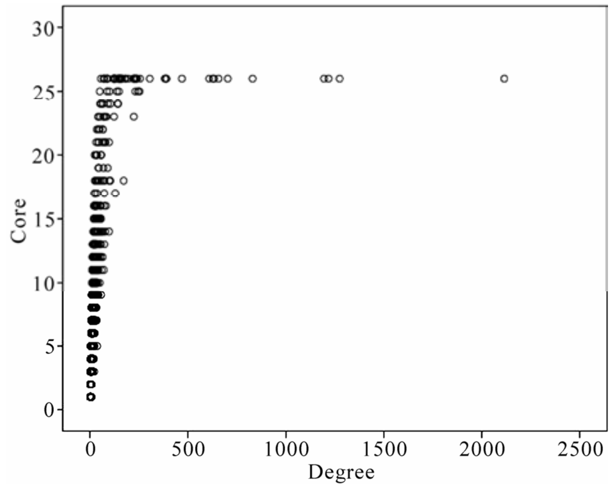
(a)



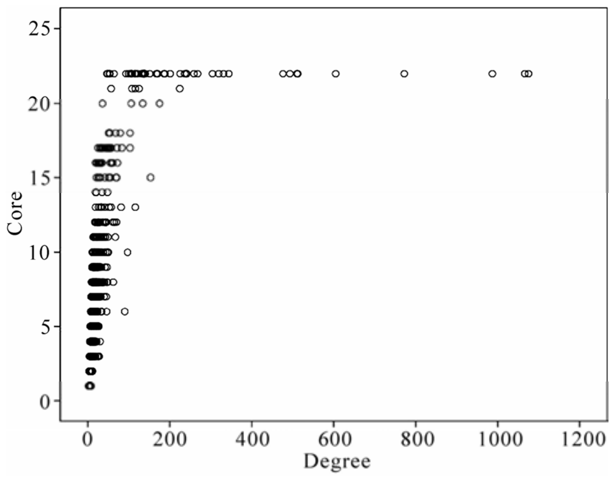
(b)



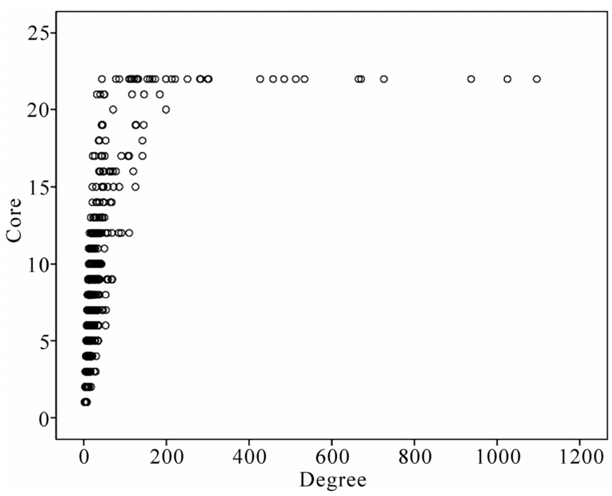
(c)



(d)



(e)



(f)

Figure 7. The relationship between the degree and the core of the nodes. (a) 2001.7; (b) 2002.7; (c) 2003.7; (d) 2004.7; (e) 2005.7; (f) 2006.7.

changes.

Figure 7 shows the set of degree in each core. X-axis is the set of degree in each core. Y-axis is the core of network in July. Core of the Internet decline with time which corresponds with the conclusion we have known in the **Figure 4**. The bottom-degree distributes in each core but top-degree only appears in the high-core. The high-core includes the nodes whose degree changes in a large span. So the change of the maximum degree has impact on the high-core in network to a certain extend. The rarefaction in the high-core and the denseness in the low-core tells us that the centralization is going to fall off.

4. Conclusions

The data we use in this paper is massive and with a large time span from July, 2001 to January, 2008. They are observed by the partner of Embedded Technology Laboratory in Northeastern University, CAIDA. The laws we get by analyzing these data are as follows:

The influence of the top-degree nodes on the other nodes decreases. The core of the Internet decreases with time. We can also say that the centralization of network is going to fall off. The nucleus status of network declines.

REFERENCES

[1] S. Zhou and R. J. Mondragon, "Structural Constraints in Complex Networks," *New Journal of Physics*, Vol. 9, No. 172, 2007, pp. 1-11.

[2] D. Y. Fu, H. Zhao and X. Ge, "AS-level Internet Topology Degree and Connectivity Analysis," *Compute Science*, Vol. 36, No. 10, 2009, pp. 104-105.

[3] L. Subramanian, V. N. Padmanabhan and R. H. Katz, "Geographic Properties of Internet Routing," *Proceedings of the USENIX Annual Technical Conference*, Monterey, 2002, pp. 243-259.

[4] J. Li, M. Sung, J. Xu, *et al*, "Large-scale IP Trace Back in High-Speed Internet: Practical Techniques and Theoretical Foundation," *Proceeding of the IEEE Symposium on Security and Privacy*, California, USA, 2004, pp. 115-129.

[5] CAIDA Skitter, [EB/OL], http://www.caida.org/tools/measurement/skitter/

[6] W. Willinger and J. Doyle, "Robustness and the Interact: Design and evolution," [EB/OL]. 2002. http://netlab.altech.edu/Internet/

[7] D. B. West, "Introduction to Graph Theory," China Machine Press, Beijing, 2006, pp. 339-348.

[8] X. F. Wang, X. Li and G. R. Chen "The Theory and Application of Complex Network," TsingHua University Press, Beijing, 2006.

J. Software Engineering & Applications, **2010, 3, 906-913**
doi:10.4236/jsea.2010.310107 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Accuracy of Measuring Camera Position by Marker Observation

Vladimir A. Grishin

Space Research Institute (IKI), Russian Academy of Sciences, Moscow, Russia.
Email: vgrishin@iki.rssi.ru

Received July 14th, 2010; revised August 10th, 2010; accepted August 14th, 2010.

ABSTRACT

A lower bound to errors of measuring object position is constructed as a function of parameters of a monocular computer vision system (CVS) as well as of observation conditions and a shape of an observed marker. This bound justifies the specification of the CVS parameters and allows us to formulate constraints for an object trajectory based on required measurement accuracy. For making the measurement, the boundaries of marker image are used.

Keywords*: *Computer Vision System, Camera Position Measurement, Marker Observation, Lower Bound to Errors

1. Introduction

CVSs are widely applied for a solution of motion control problems. This fact is associated by the following conditions. First, the computational capability of available processors allows for the real-time processing of large volumes of information formed by TV cameras. The information processing time proves to be acceptable to a number of practical problems [1-6]. Second, the increasing application of computer-aided control systems of unmanned aerial vehicles requires the enhancement of the vector of measured parameters to solve the automatic landing problem [5]. Another task is a docking problem (including the spacecraft docking), which requires precise measuring a relative position for solving the terminal control task [6]. As an example we can refer to the docking the first European Automated Transfer Vehicle (ATV) “Jules Verne” to the International Space Station (ISS) on 3 April 2008. In the above experiment, a special computer vision system was used for measuring the relative spatial and angular position.

All of these facts stimulate interest in estimation of the potential accuracy (lower bounds to errors) of measuring the position parameters as a function of the marker shape, its observation condition and technical parameters of the CVS. This allows us to evaluate an applicability of CVSs to solving control problems under specific conditions as well as to optimize the CVS parameters from the viewpoint of ensuring the required accuracy of measurements. There are a small number of publications devoted to the problems of determining the current coordinates measurement precision estimation. Most publications are based on experimental approach (full-scale experiments or stochastic simulation) to the measurement precision estimation. For obtaining reliable estimation, such approach requires too much time and additionally the full-scale experiments are very expensive.

In [7], the Cramér–Rao bound is constructed to camera position estimation by docking marker observation. For position estimation, a set of the marker features (points of interest) are used, namely corners, contrast spots and others. This approach is suitable for the case of small or medium marker observation distance. In such distances the visible size of marker is of order tens or hundreds of pixels in any direction. In the present paper, we consider the approach, which is suitable for large distances by using the boundaries between marker image and background. This approach allows obtaining lower bound to errors of measuring object position with small computational expenses. It allows in one’s turn to optimize CVS parameters and marker shape for a specified set of the observation conditions.

In Section 2, we formulate the assumptions for constructing the bound to errors. In Section 3, we construct a Cramér–Rao bound to the measurement errors and, in Section 4, we present experimental results.

2. Assumptions Made When Constructing a Bound

We make the following simplifying assumptions to esti-

mate the methodic errors:
- The resolution of the optical system is the same over the frame area.
- There are no geometrical distortions of the optical system (or they are compensated for during the pre-processing of images).
- The optical system is calibrated during its manufacturing and the calibration error is negligible.
- The exposure time tends to zero, so smearing of the picture due to the motion of the object during shooting can be neglected.
- The precision of marker localization is limited by signal to noise ratio.
- The parameters of this noise law are the same over the area of a frame.
- The pixel size of CCD matrix tends to zero.

All of these assumptions, except for the last one, are quite easily realizable at moderate cost. In regard to the last assumption, it is introduced for simplification of analysis. Without this simplification, an analytical solution is very difficult. Apparently, it is possible to obtain some asymptotic estimations of additional object position measurement errors, which is conditioned by limited size of CCD matrix pixels. In any case, this problem should be a subject of separate analysis. Thus, the used model has no error sources except for the image noise.

3. Cramér–Rao Bound to Measuring Errors

The construction and application of a likelihood function and Cramér–Rao bound for measurement errors are extensively described in the literature [8-10] and others. A likelihood function is used for constructing the Cramér–Rao lower bound to the variance of estimated parameters. The schematic view of the marker shooting is shown in **Figure 1**. The marker is placed in the coordinate's origin.

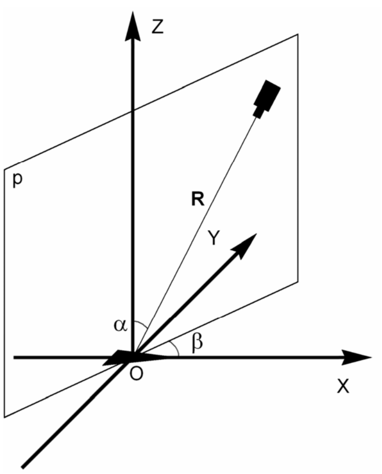


Figure 1. TV camera position.

The optical system forms the image of observed marker in the plane of a CCD matrix. The space position of the TV camera and its orientation gives a vector of parameters **A** that should be estimated. Camera coordinate system is shown in the **Figure 2**. Projection center C of the camera is placed on the end of vector **R** (**Figure 1**), which is turned with respect to a normal of the surface of marker on angle α in the plane p which pass through axis OZ and is preliminarily rotated on the azimuth on angle β relatively plane XOZ.

In the initial camera position vectors $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ are given by the coordinates as follows:

$$\mathbf{e}_1 = (0,0,-1)$$
$$\mathbf{e}_2 = (-1,0,0)$$
$$\mathbf{e}_3 = (0,1,0).$$

The above three vectors are rotated by an angle α together with the projection center of camera C in the plane p. So, the obtained coordinates of the vectors are the following:

$$\mathbf{e}_1 = (e_{11}, e_{12}, e_{13})$$
$$\mathbf{e}_2 = (e_{21}, e_{22}, e_{23})$$
$$\mathbf{e}_3 = (e_{31}, e_{32}, e_{33}).$$

Let γ, φ and ψ be three rotation angles around the vectors $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ respectively. The first rotation is the rotation by the angle γ. Since the TV camera is space stabilized so that, the image of observed marker is in the center of the vision area, it is possible to suppose the angles φ and ψ small enough ($\varphi \approx 0$, $\psi \approx 0$). Hence, the rotation operators by the angles φ and ψ are approximately commutative.

The coordinates of any *i*-th marker point (X_i, Y_i, Z_i) taken in camera coordinate system are the following:

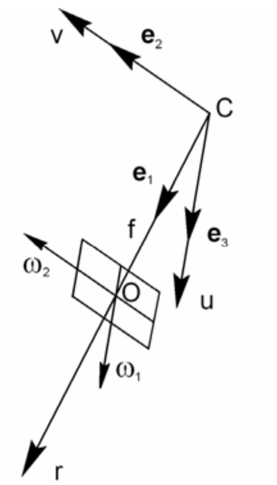


Figure 2. Camera coordinate system.

$$X_i' = X_i - X$$
$$Y_i' = Y_i - Y$$
$$Z_i' = Z_i - Z,$$

where (X, Y, Z) is the coordinates of camera projection center C. The coordinates of the *i*-th point of the marker in CCD matrix are calculated by:

$$a = \frac{f}{X_i' e_{11} + Y_i' e_{12} + Z_i' e_{13}}$$
$$\omega_1^i = a\left(X_i' e_{31} + Y_i' e_{32} + Z_i' e_{33}\right)$$
$$\omega_2^i = a\left(X_i' e_{21} + Y_i' e_{22} + Z_i' e_{23}\right),$$

where f -is a focal distance of the camera.

For the specified camera's spatial and angular positions, the i-th point (X_i, Y_i, Z_i) taken in the coordinates of CCD matrix depends on the parameters:

$$\omega_1^i = \omega_1^i\left(X, Y, Z, \gamma, \varphi, \psi\right)$$
$$\omega_2^i = \omega_2^i\left(X, Y, Z, \gamma, \varphi, \psi\right).$$

Since we consider an observation of marker from medium and long distances, the measurement angular errors of φ and ψ as well as the translation errors in the direction of the vectors $\mathbf{e}_3$ and $\mathbf{e}_2$ are heavily correlated. So, we estimate the precision only for four parameters, that are given by a vector $\tilde{\mathbf{A}} = (r, v, u, \gamma)$. Axis *r* is parallel to $\mathbf{e}_1$, *v* is parallel to $\mathbf{e}_2$ and *u* is parallel to $\mathbf{e}_3$.

The construction and application of the likelihood function are well known from [8-10] and others. This likelihood function is used for constructing the Cramér–Rao lower bound to the variance of estimated parameters. The likelihood function depends on parameters being under estimation. The estimations of the parameters are defined by the values that provide the extremum of the likelihood function:

$$P\left(\tilde{\mathbf{A}}\right) \to extr,$$

where $P\left(\tilde{\mathbf{A}}\right)$ is the likelihood function.

The necessary condition of extremum is given by:

$$\frac{\partial P\left(\tilde{\mathbf{A}}\right)}{\partial A_i} = 0, \quad i = 1, \ldots, 4.$$

Accordingly, we can use a logarithm of the likelihood function for finding of extremum of $P\left(\tilde{\mathbf{A}}\right)$. Analogous condition of extremum can be:

$$\frac{\partial \ln P\left(\tilde{\mathbf{A}}\right)}{\partial A_i} = 0, \quad i = 1, \ldots, 4.$$

Covariance matrix of estimated parameters is:

$$R = J^{-1},$$

where J is the Fisher information matrix, which is calculated from the likelihood function. According to the Cramér–Rao inequalities, the lower bounds to the variances of unbiased estimation errors are given by:

$$\sigma_r^2 \ge R_{11}\left(\tilde{\mathbf{A}}\right), \ \ \sigma_v^2 \ge R_{22}\left(\tilde{\mathbf{A}}\right),$$
$$\sigma_u^2 \ge R_{33}\left(\tilde{\mathbf{A}}\right), \ \ \sigma_\gamma^2 \ge R_{44}\left(\tilde{\mathbf{A}}\right).$$

We estimate the covariance for the estimation of vector **A**. For this goal, we first determine the Fisher information matrix, which is expressed via the second derivatives of the likelihood function as follows:

$$J_{ij} = -E\left[\frac{\partial^2 \ln P\left(\tilde{\mathbf{A}}\right)}{\partial A_i \partial A_j}\right] = E\left[\frac{\partial \ln P\left(\tilde{\mathbf{A}}\right)}{\partial A_i}\frac{\partial \ln P\left(\tilde{\mathbf{A}}\right)}{\partial A_j}\right],$$

where $E[\ldots]$ is a mathematical expectation.

Let's consider an observed image of marker:

$$\mu(\boldsymbol{\omega}) = s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right) + \xi(\boldsymbol{\omega}),$$

where $s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right)$ is the marker image and $\xi(\boldsymbol{\omega})$ is noise with intensity $\sigma^2 = N_0/2$. Without loss of commonness, we can suppose that a brightness value of marker image $s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right)$ is equal to one, and a brightness of remaining part of the cadre is zero.

In [11], an expression of Fisher Information Matrix was derived for the case of one-dimensional signal. For the two-dimensional case, this expression can be easily obtained by the same way:

$$J_{ij} = \frac{2}{N_0} E\left[\int_\Omega \frac{\partial s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right)}{\partial A_i}\frac{\partial s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right)}{\partial A_j} d\boldsymbol{\omega}\right],$$

where Ω is a marker image area and $d\boldsymbol{\omega}$ is an elementary square in Ω.

In general case, the calculation of the Fisher information matrix requires to determine the above mathematical expectation $E[\ldots]$. In our case, the expression in square brackets is deterministic, and therefore we obtain the following elements of the Fisher information matrix:

$$J_{ij} = \frac{2}{N_0}\int_\Omega \frac{\partial s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right)}{\partial A_i}\frac{\partial s\left(\boldsymbol{\omega}, \tilde{\mathbf{A}}\right)}{\partial A_j} d\boldsymbol{\omega} \qquad (1)$$

Let's consider derivatives. The $\tilde{\mathbf{A}}$ is the vector of parameters that gives the camera position. The finite difference approximation of the derivative is defined as follows:

$$\frac{\partial s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)}{\partial A_i} \approx \frac{\Delta s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)}{\Delta A_i} = \frac{s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}+\Delta\tilde{\mathbf{A}}_i\right)-s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)}{\Delta A_i},$$

$$\Delta\tilde{\mathbf{A}}_i = \left(0,....\Delta A_i,....0\right).$$

Figure 3 shows the marker image in the initial position $s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)$. In **Figure 4**, the marker images are shown for both the shifted position $s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}+\Delta\tilde{\mathbf{A}}_i\right)$ and the initial position $s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)$. The gray colours of different intensity are used for marking difference between both these images.

The difference can be calculated by integrating an optical flow on the contour of marker as follows:

$$\frac{\partial s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)}{\partial A_i} = \begin{cases}\left(\mathbf{n},\mathbf{Q}_i\right), & \boldsymbol{\omega}\in C\\ 0, & \boldsymbol{\omega}\notin C,\end{cases}$$

where $\mathbf{n}$ is the external normal ($\|\mathbf{n}\|=1$) with respect to the marker image boundary (contour), $\mathbf{Q}_i$ is the optical flow, which is caused by $\Delta\tilde{\mathbf{A}}_i$, $\left(\mathbf{n},\mathbf{Q}_i\right)$ is a scalar product of the vectors $\mathbf{n}$ and $\mathbf{Q}_i$ and C is the marker boundary. In such a way, we show that the surface integral (1) is reduced to the following contour integral:

$$J_{ij} = \frac{2}{N_0}\int_C \left(\mathbf{n},\mathbf{Q}_i\right)\left(\mathbf{n},\mathbf{Q}_j\right)dl \qquad (2)$$

Thus we have obtained the expression for any element of the Fisher information matrix. For the one segment, the integral (2) can be numerically calculated, for instance, by the trapezium method:

$$p_{ij} = \frac{2}{N_0}\frac{1}{2}\sum_{n=1}^{N-1}\left(\left(\mathbf{n},\mathbf{Q}_i^n\right)\left(\mathbf{n},\mathbf{Q}_j^n\right)+\left(\mathbf{n},\mathbf{Q}_i^{n+1}\right)\left(\mathbf{n},\mathbf{Q}_j^{n+1}\right)\right)\delta l \qquad (3)$$

Figure 5 explains the calculation of scalar product $\left(\mathbf{n},\mathbf{Q}_i\right)$. The calculation of $\left(\mathbf{n},\mathbf{Q}_j\right)$ is made similarly. The δq is the difference between scalar product $\left(\mathbf{n},\mathbf{Q}_i\right)$ in the integral (2) for this segment, and in the expression (3) for this segment. Notice that δq is proportional to $\left(\Delta A_i\right)^2$ and tends to zero in condition of $\Delta A_i \to 0$. So we can neglect this term. Calculation of the expression (3) should be performed for all sections of the maker boundary.

4. Experimental Results

To illustrate the application of the obtained relations, we estimated the errors of calculating position parameters for the marker shown in **Figure 6**. The marker is given by the isosceles triangle. The base of the triangle equals to two meters and its height equals to three meters. The triangle has the round spot in his centre. Contour (boundary) C of this marker includes both external boundary of this triangle and internal boundary of the spot in the triangle centre.

Let's specify the following camera parameters. The focal distance of the optical system is 18 mm. The field of camera view is $23.23°\times 23.23°$. Errors of position are calculated for a set of values of angle α: $\alpha = 5, 15, 25, 35, 45, 55, 65^\circ$ (7 values) and set of values of angle β: $\beta = 0, 10, 20,......350^\circ$ (36 values). The distance of the marker observation is $r = 50$ м. We put a noise intensity to be equal to 0.2 ($\sigma = 0.2$).

Figure 7 shows the calculation results for the mean square errors of coordinates and normalized correlation. The coordinates are measured in meters and the values of angles are measured in degrees. The errors are given by the appropriate surfaces over the matrix of size 7 × 36 samples, where the matrix sizes are determined by the sets of α and β values respectively.

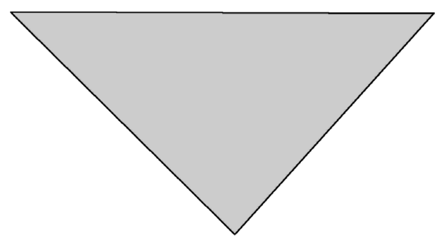


Figure 3. Triangle marker.

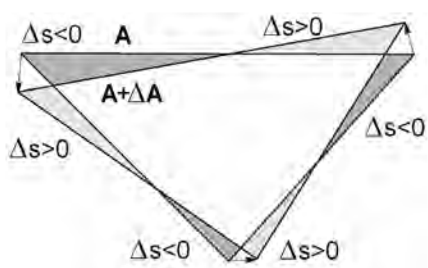


Figure 4. Difference $s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}+\Delta\tilde{\mathbf{A}}_i\right)-s\left(\boldsymbol{\omega},\tilde{\mathbf{A}}\right)$.

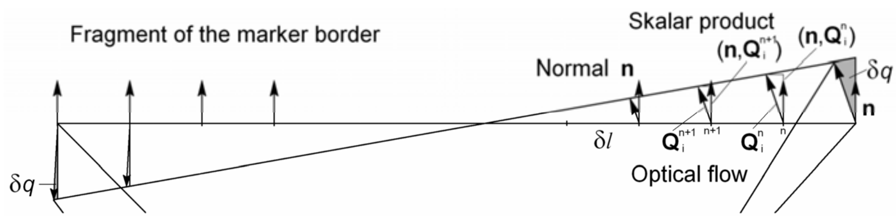


Figure 5. Calculation of the scalar product on the one segment of marker boundary.

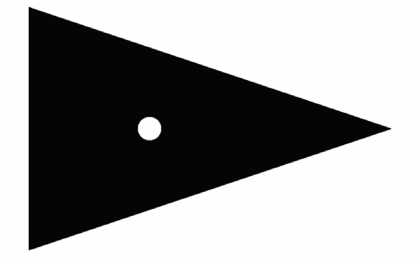
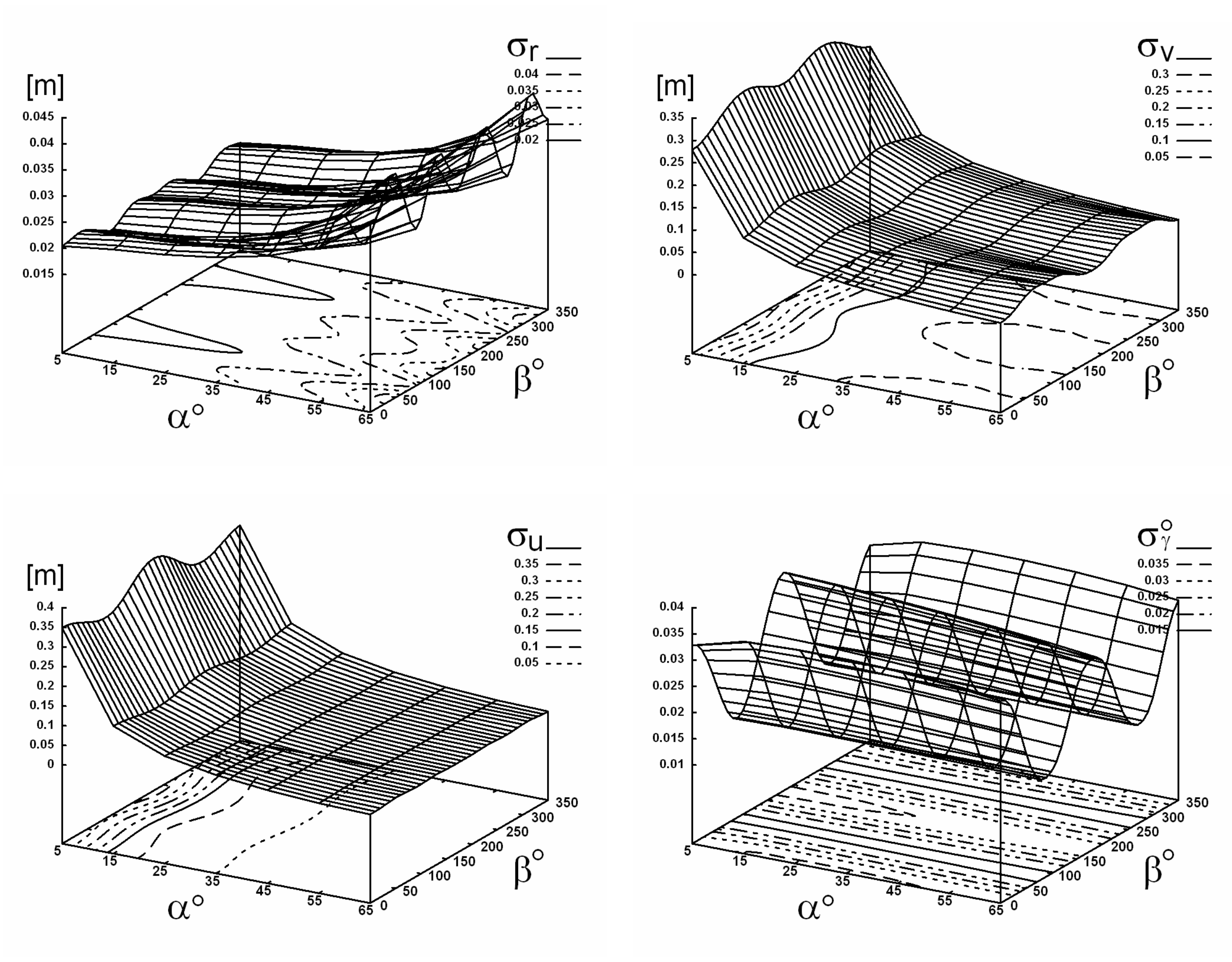


Figure 6. Marker shape.

Accordingly to **Figure 7**, for the distance of 50 m and the noise intensity $\sigma = 0.2$, the range $|\mathbf{r}|$ can be measured with error $\sigma_r = 0.02 - 0.04$ m, as well as the displacement in a CCD matrix plane can be measured with errors $\sigma_V, \sigma_U = 0.05 - 0.4$ m. Rotation around the vector $\mathbf{r}$ can be measured with the error $\sigma_\gamma = 0.015 - 0.04^\circ$. As followed from **Figure 7**, the functional dependences of measurement errors and normalized correlation of linear and angular coordinates are very complicated functions. We have considered the maker of uniform brightness. In this case, only the contrast boundary operates in the marker image. The calculated precision values are much higher than the similar values in [7] that are based on using a small set of features (points of interest) of the marker. Using the boundaries of marker image for measurement provides an increase of the measurement precision. Mention should be made that optical system distortions and low resolution of CCD camera can seriously deteriorate the precision of measurement. Joint analysis of noise and camera resolution influence on the precision of measurement is complicated enough.

The above values of the mean square error and the normalized correlation should be taken in an account in creating the computer vision system. The significant values of the normalized correlation show the considerable dependences between control loops of object position coordinates. This fact should be taken into account in the control system. The development of a computer vision system should be carried out together with the development of the marker shape.



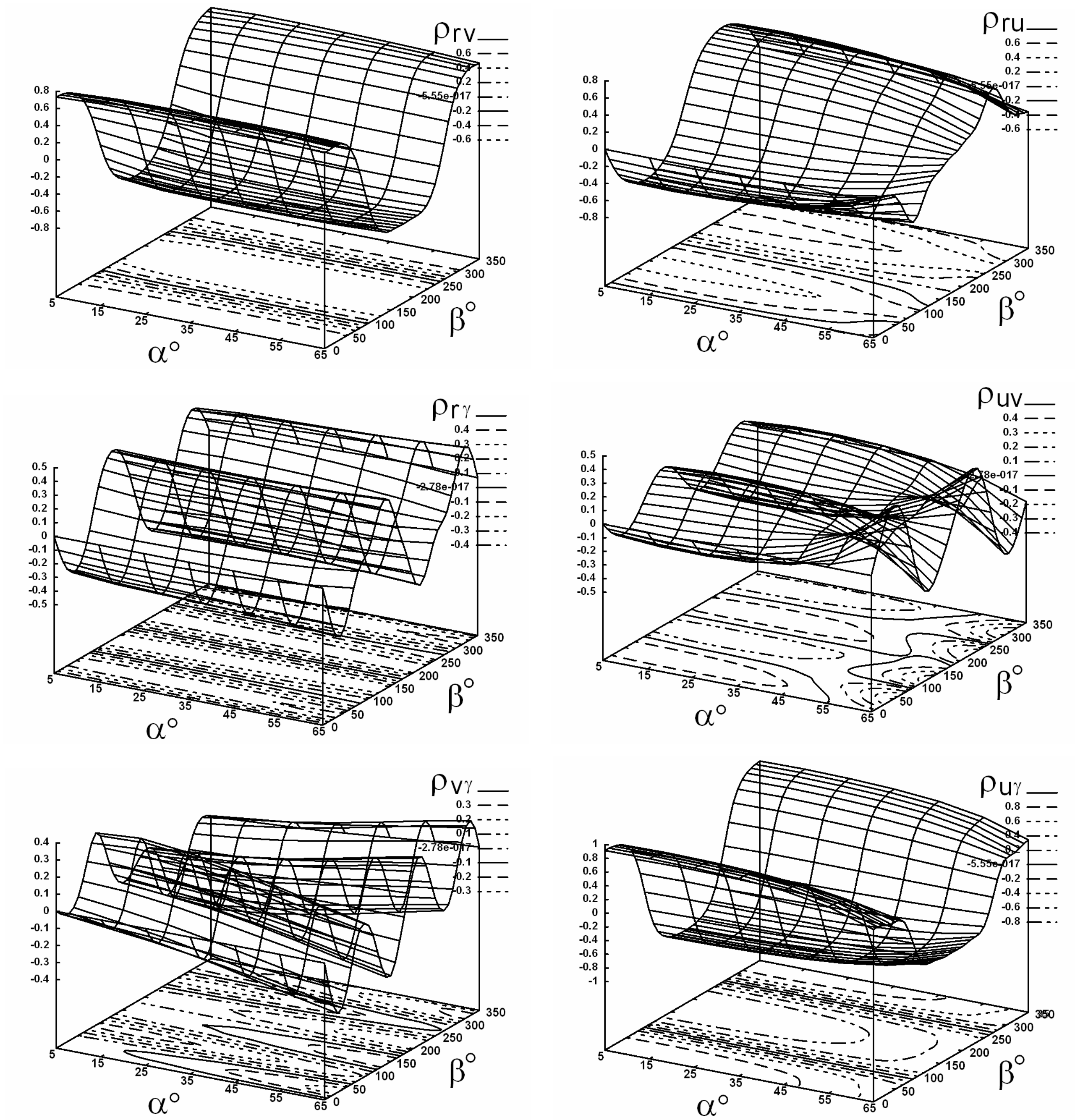


Figure 7. Errors of estimated parameters and correlation bonds between them (normalized correlation).

For comparison, we estimated the errors of calculating position parameters for the T-shaped marker shown on **Figure 8**.

This marker has the same area as the marker on **Figure 6**. **Figure 9** shows the calculation results for the measurement error of coordinates.

Accordingly to **Figure 9**, the T-shaped marker provides a slightly higher precision of position parameters' measurement.

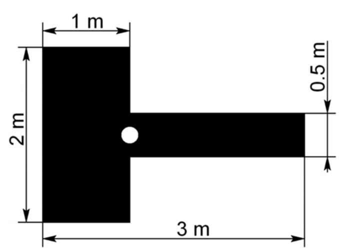


Figure 8. T-shaped marker.

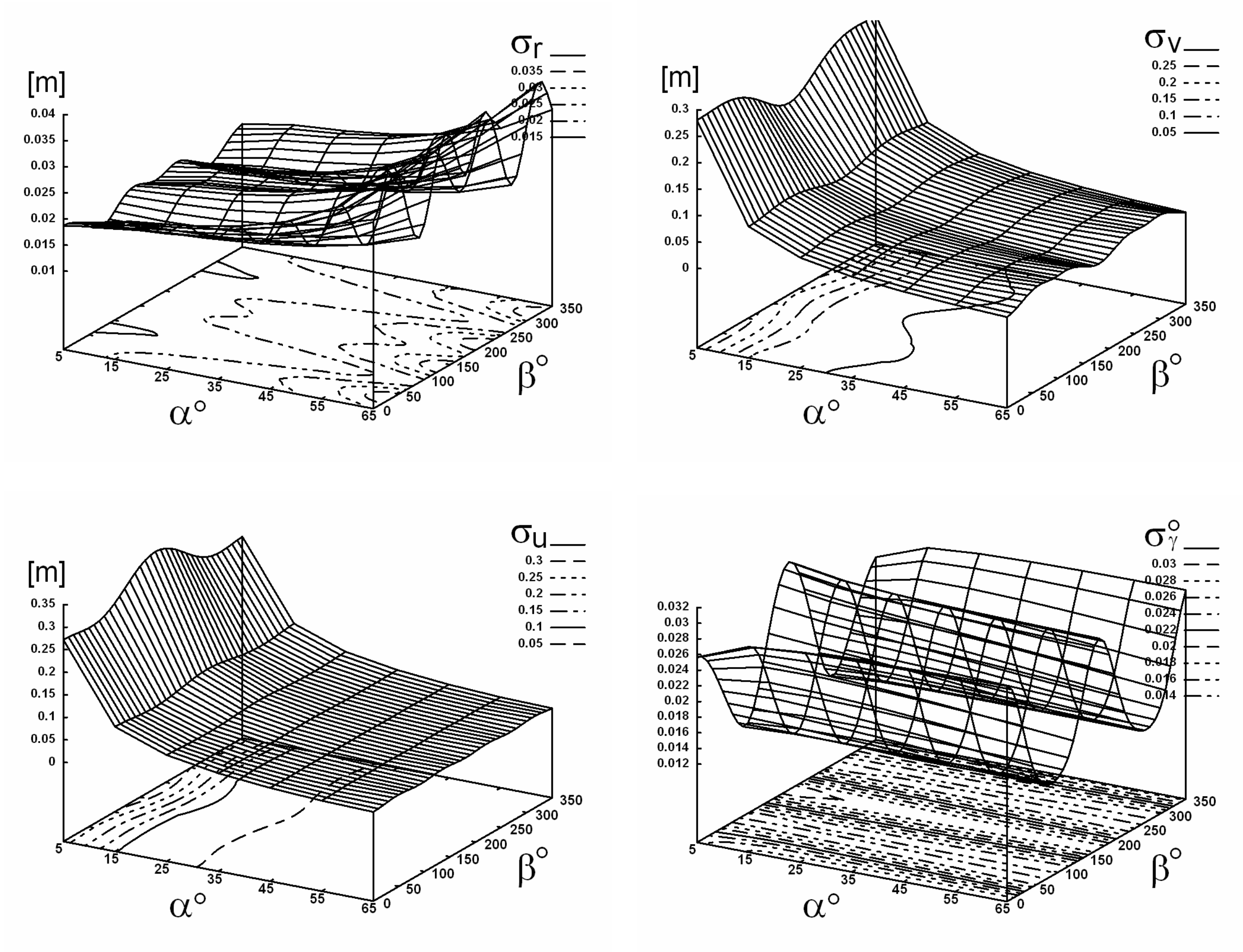


Figure 9. Errors of estimated parameters for T-shaped marker.

5. Conclusions

The new method has been proposed for estimating the errors of determining the TV camera position. This method is based on using the marker image of a given shape. The method allows us to estimate the measurement errors depending on shooting conditions and CVS parameters. The obtained error's estimations are useful for development of CVSs and particularly for optimization of their parameters.

6. Acknowledgements

This work was supported by the Russian Foundation for Basic Research, project no. 09-01-00573-a

REFERENCES

[1] C. De Wagter and J. A. Mulder, "Towards Vision-Based UAV Situation Awareness," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, San Francisco, California, August 2005, pp. AIAA-2005-5872.

[2] S. Saripalli, J. F. Montgomery and G. S. Sukhatme, "Visually-Guided Landing of an Unmanned Aerial Vehicle," *IEEE Transactions on Robotics and Automation*, Vol. 19, No. 3, June 2003, pp. 371-380.

[3] S. Saripalli and G. S. Sukhatme, "Landing a Helicopter on a Moving Target," *Proceedings of IEEE International Conference on Robotics and Automation* (*ICRA*'2007), Roma, Italy, April 2007, pp. 2030-2035.

[4] T. Kubota, S. Sawai, T. Misu, T. Hashimoto, J. Kawaguchi and A. Fujiwara, "Autonomous Landing System for MUSES-C Sample Return Mission," *Proceedings of the Fifth International Symposium on Artificial Intelligence, Robotics and Automation in Space* (*ISAIRAS*'99), ESA SP-440, Noordwijk, The Netherlands, June 1999, pp. 615-620.

[5] C. S. Sharp, O. Shakernia and S. S. Sastry, "A Vision System for Landing an Unmanned Aerial Vehicle," *Proceedings of IEEE International Conference on Robotics and Automation* (*ICRA* 2001), Seoul, Korea, May 2001,

pp. 1720-1727.

[6] C. Akinli, “Semi-Autonomous Terminal Phase Spacecraft Docking Attitude Determination and Control,” 2004. http://cengiz.akinli.org/research/vsgc-2004.pdf

[7] V. A. Grishin, “Precision Estimation of Camera Position Measurement Based on Docking Marker Observation,” *Pattern Recognition and Image Analysis*, Vol. 20, No. 3, 2010, pp. 341-348.

[8] H. Cramer, “Mathematical Methods of Statistics,” Princeton Univ. Press, USA, 1946.

[9] I. A. Ibragimov and R. Z. Khasiminskii, “Asymptotic Estimation Theory,” Nauka, Moscow, 1979.

[10] A. B. Kryanev and G. V. Lukin “Mathematical Methods for Processing Indeterminate Data,” Fizmatlit, Moscow, 2003.

[11] H. L. Van Trees, “Detection, Estimation, and Modulation Theory,” Part 1: “Detection, Estimation, and Linear Modulation Theory,” John Wiley & Sons Inc., New York, 2001.

J. Software Engineering & Applications, **2010, 3, 914-925**
doi:10.4236/jsea.2010.310108 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

BOSD: Business Object Based Flexible Software Development for Enterprises

Jindan Feng, Dechen Zhan, Lanshun Nie, Xiaofei Xu

Research Center of Intelligent Computing for Enterprises and Services, Harbin Institute of Technology, Harbin, China.
Email: Jindan.Feng@gmail.com, {dechen, nls, xiaofei}@hit.edu.cn

Received July 15th, 2010; revised August 3rd, 2010; accepted August 15th, 2010.

ABSTRACT

The enterprise software need adapt to new requirements from the continuous change management. The recent development methods have increased the flexibility of software. However, previous studies have ignored the stability of business object and the particular business relationships to support the software development. In this paper, a coarse-grained business object based software development, BOSD, is presented to resolve this problem. By analyzing the characteristics of variable requirement, business objects are abstracted as the separately-developed unit from business process, and are assembled to system through their relationships. The methodology of BOSD is combined with MDA (Model Driven Architecture) and implemented on the semiautomatic platform.

Keywords: *Business Requirement Change, Business Object, Relationship, Business Process, Information Systems, Flexibility, MDA*

1. Introduction

Enterprises use information system to optimize their management, and further enhance its competitive abilities in the markets. There are many changes in the optimization processes, such as the transformation from extensive management to intensive management, the transformation from various objects to uniform business object, and the transformation from disordered process to normative process [1]. One of the major challenges in the enterprise software is the adaptability at run time.

Currently, there are basically two types of factors to flexible software development. The first factor includes the potential changes of requirements and the disciplinarian of these changes. The researchers consider that the software adaptabilities focus on the business processes [2,3], and the processes follow the special change patterns [4]. The second factor is the proper development approaches which support the disciplinarian. There are three representative approaches to achieve the flexible software: Component-Based Software Development (CBSD), Model Driven Development (MDD), and Service-oriented Software Development (SOSD).

The CBSD emphasizes the software architecture, on which the software is iteratively assembled with different grained components [5]. The software developed by CBSD adapts to the changes of environment by reusing the existing components [6]. The MDD is an approach that generally separates the functionality from the implementation from the perspective of separation concern. The Computation-Independent Model (CIM), which is composed of business-centric models, is transformed to Platform-Independent Model (PIM) which represents software design rather than the details on the technology platform. The PIM is further refined to Platform-Specific Model (PSM), which is implementation model on the concrete platform. Finally, the software corresponding to the PSM is produced by code generator [7]. The MDD elevates the abstract level from codes to models. It makes the software quickly adapt to the new requirements through modifying the models. The approach depends on two aspects, the first, the well-formed models which have a directly effect on the correctness of the software [8]; the second, the mature technologies of the mapping and transformation among models [9]. However, the MDD cannot be separately used, and must be combined with the suitable modeling methodology to realize the specification of enterprise software [10]. The SOSD proposes that the service has a larger granularity than the source code traditionally, and the service is considered fundamental elements for developing applications [11]. The Business Process Execution Language (BPEL), which defines the sequence of Web Services Description Language (WSDL) interface, provides support for executable and abstract business process. Hence, the web service can be dynamic and flexible to meet changing business needs.

The above mentioned approaches are essentially classified into two categories: business process oriented software development approach and business object oriented software development approach [12].

The business process describes the ordering of activities for the purpose of achieving business objectives in the context of business organization and policy [13]. The business process oriented software development is an approach that focuses on the identifying business activities and decomposes the interaction of these business activities. A business process is performed by cooperation of a number of business resources called business objects. For an explicit business objective, these business activities are implemented and encapsulated into business objects according to the relationships of their functions [14]. The software based on business process can support dynamic configuration of activities with specific control styles such as the workflow [15].

The business object oriented software development is an approach that identifies business objects and the relationships between them. And then the business objects are assembled into the software as loosely coupled systems. A further benefit is that when the business process changes, the software can rapidly reconfigured through reusing the existing business objects to meet the changes at reduced cost of development and modification [16].

Although the approaches are considered business process-centric and business object-centric, respectively, they are complementary to the design of software, and always are used together [4,17,18]. In the combination of the processes and objects, one of them is the more changeable. As above mentioned, the business objects have more stability than the process. However, there are many different definitions of business object proposed presently. The general definition comes from Domain Task Forces (DTF) of Object Management Group (OMG). The Business Object is defined as a representation of a thing active in the business domain, including at least its business name and definition, attributes, behaviors, relationships, rules, policies, and constraints. A business object may represent, for example, a person, place, event, business process, or concept. Typical examples of business objects are: employee, product, invoice, and payment [19]. The business object can be used to describe the meta-model of enterprise [20]. A business object is also seen as a "class" or a set of classes in the object-oriented system, and further support a defined business area [21]. The existing definition of business object generally is considered any objects which are the inputs and outputs of business actions. The business objects generally refer to the classes which cover the contents in the different domains, such as business domain and software domain. In fact, the objects play different roles in the phases of business modeling, software design and software implementation etc. Therefore, business objects need to be distinguished explicitly.

In this paper, the concepts of business object and their relationships are proposed for the designing of enterprise software. We present (1) an explicit definition of business object; (2) the semantic completeness of business object; (3) business object is considered the smallest units of the business modeling, and the biggest units of the software realization; (4) a business component is an implementation of business object on the technology platform, and is a coarse-grained component.

In short, there lack of the strict and systemic methods to support the development of flexible software for enterprise. The potential requirements which occur in application phase need to be considered adequately in the software design phase. Aiming to adapt the more changes in the field of business management, the software adjusts itself locally by right of the reusability. The paper proposes an approach of software development based on business object, which is combined with the excellences of the mentioned approaches (CBSD and MDD). The approach is divided into three steps. First, we identify the business objects which have relative stability, and are regarded as separately-developed units of software. Second, we analyze the relationships among the business objects. Last, the business objects are composed into the flexible software under the control of the relationships mechanisms. The approach combined with the Model Driven Architecture (MDA) should be supported by a semiautomatic development platform. The paper attempts to provide the methodology for business modeling and its realization project for flexible software.

The structure of this paper is as follows. In Section 2, the variable requirements are briefly introduced. In Section 3, we present a definition of business object and the relationships between them. In Section 4, we further discuss the modeling methodology based on business object, and illustrate the details in our approach. The conclusion is given in Section 5.

2. The Variable Business Requirement

The changes of business requirements bring some negative effects on enterprise software at run time, sometimes even fatal impacts. For instance, 1) if a new data item is added and considered the primary key of the data view in the database, the software must adjust the relational components to adapt the change; 2) If the dependence relationships, (such as the state sequence of business documents, the associations between the documents), are changing in business system, the business process reengineering work through information technology is also needed for the new relationships. Therefore, if the software has abundant adaptability and flexibility, it can enhance the reengineering efficiency and decrease the reuse cost. In order to develop successfully flexible software for enterprise, it's primary problem for the software developers to understand accurately the business requirements and their changeable characteristics at run time.

In the real enterprises, business requirements always changes continually, and the changes follows the particular rules. According to our experiences in software development, we find that there are various changes of the business process between the different enterprises, which are in the same industry. In this section, we illustrate the variable requirements using the Purchase Order as an example. There are the different processes of making the Purchase Order, and every process represents a sequence of creating Purchase Order in **Figure 1**. The general routes is "submit Requirement → schedule Plan → inquire Price → build Order" in Process 1. The optional route shows that Purchase Order is refined directly from the Requirement Bill in Process 2, and the route is used for emergency requirements. With the improvement of information degree in enterprise system, the contents of Purchase Requirement in Process 4 can be imported from the records of the material requirements in the workshops, or obtained from computing the replenishment amount for inventory. The different types of Purchase Requirements, such as large facilities and production materials in Process 4, will go through the different approval processes.

Although there are different processes which are used for creating the Purchase Order, the business documents in the processes are limited to a certain scope, and they are Requirement, Plan, Quotation Invoice and Order for purchase. And then, the corresponding forms in each Process have similar kernels, for example, the every Requirement Bill contains the data items as follows: keys, Need Department, Item Code, Need Amount and Date etc. Sometimes, these data items may have the different names, but the same semantics. According to the standardization degree, the assistant information in documents may exist or not, such as the data items used for describing state transformation. Therefore, we find that the documents are similar, and there are little differences between them in the same business transactions.

From the analysis above we can know, the processes of making purchase order are various, and essentially the relationships between the documents are various. It's obvious that the business documents have relative stability. The business documents are more stable than the business processes. By analyzing the feature of business process, we find that the processes also follow the rules. The relationships between business objects are divided into three types: approval process, association process and integration process, and they are shown in **Figure 2**. The approval process is one of the basic processes in the lifecycle of business documents. The approval process is a set of activities, including auditing and affirming. The people at the different ranks run the activities corresponding to their rights. The association process is composed of the activities which deal with the many-to-many relationship between two documents. The integration process is an activity set to create new documents through the computation based on one or more existing documents. The computation rules are always complicated, and need to be produced by hands. If these relationships are identified clearly, they are implemented based on the particular patterns. Therefore, the difficulty of assembly is reduced to a certain degree.

The business system is thought as dynamic network,

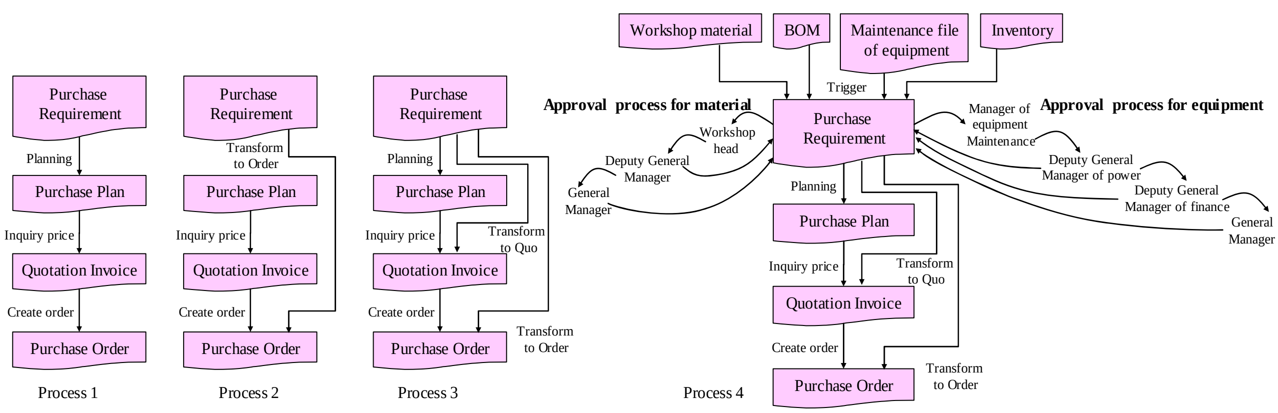


Figure 1. The optional processes of creating purchase order.

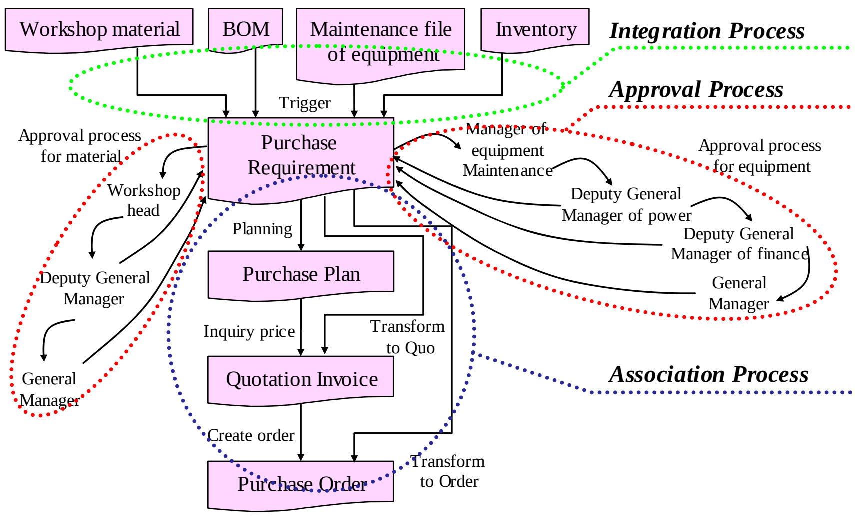


Figure 2. Three types of business processes.

and is composed of stable business documents and the various business processes. If the software has been implemented only using the approach based on the process, it is difficult to adapt quickly to the new relationships. Therefore, the approach is combined business objects with business processes together in this paper. The process-oriented requirement analysis is transformed into the object-oriented software development in the approach. First, requirement analysts understand the business processes in the enterprise, and identify the business documents. Second, the business objects are abstracted from the documents by software designers. The objects are as independent as possible, thus they can be reused for the variable processes. Third, the connections between objects are built in the light of current requirements. Finally, the business objects are realized to business components. The business processes are transformed to the cooperation relationships among the components, which are supported by the workflow engine. By doing so, we enable to assemble the flexible software form the business components and workflow engine.

3. Business Object

Basing on the business object from OMG, the business documents and their relationships are identified to business object and these three relationships in **Figure 3**, respectively. And then, we separate the business semantics of business object from its implementation.

Hence, our business objects differ from the objects of OMG in the following aspects: 1) from the perspective of software design, business object is a basic operation unit of system, is a integration of the data and the operations on them; 2) from the view of external application, business object is an integrated body, including the informa-

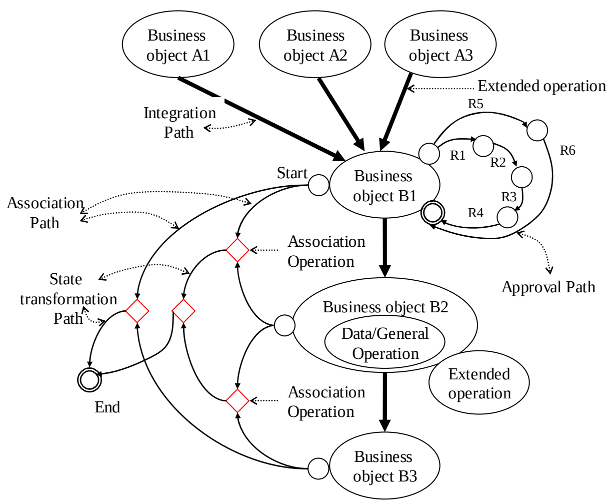


Figure 3. The business objects and their relationships.

tion which are collected and transformed by the users, the state transformations and operations which affect on the information, and the state transformation is such as start, running, end; 3) from the view of object-oriented structure, business object has an identifier number, data sets, operation sets, and its own lifecycle; 4) from the view of representation format, the business object represents that some children tables are attached to a father table; 5) from the software implementation view, business object is implemented to business component with the software technology.

The definition of business object is as follows:

Definition 1: ***Business Object*** **(*BO*)** is defined as a basic operation unit with the integral semantics in the field of enterprise business. Business object is described as *BO* = (*BOID*, *DS*, *OPS*, *SS*), where:

1) *BOID* is the *ID* number of *BO*;

2) *DS* is the data sets. *DS* = {*CDS*, {*ADS*}}; *CDS* is the core data set; *ADS* is assistant data sets. Every data set is composed of numerous data elements, which have the same or similar semantics. *DS* need satisfy the following constraints: ① $CDS \neq \varnothing$; ② *d* is a data element of the *DS*. $\forall BO, \exists X \subseteq CDS, X = \{d_1, d_2, d_3, \ldots\}$. But $\forall X$, there doesn't exist more than one x_i, $x_i \in X$, which uniquely determines the *CDS* of *BO*; ③ $ADS = \{ADS_1, ADS_2, \ldots, ADS_i, \ldots, ADS_n\}$, $ADS_i = \{d_n \mid d_n \text{ is data element}, d_n \in DS\text{-}CDS\}$ and $ADS_i \cap ADS_j = \varnothing$, where $i, j = 1, 2, 3\ldots$; ④ $CDS \cap ADS = \varnothing$; ⑤ If $ADS = \varnothing$, there is $DS = CDS$;

3) *OPS* is the operation sets of *BO*. $OPS = \{GPS, EPS\}$, $GPS \cap EPS = \varnothing$, $GPS \neq \varnothing$. *GPS* is general operation set, *EPS* is extended operation set. An operation set is composed of many associated operations. $GPS = \{op_1, op_2, \ldots, op_i, \ldots, op_n\}$, where $i = 1, 2, 3\ldots$; *OPS* need satisfy the follow constraints: ① $GPS \neq \varnothing$; ② *op* is a operation element of the *OPS*. $\forall BO, \exists Y \subseteq GPS, Y = \{op_1, op_2, op_3, \ldots\}$. $\forall Y$, there exists single one $y_i \in Y$, *that* makes *GPS* exist uniquely; ③ If $EPS = \varnothing$, then $OPS = GPS$; ④ *GPS* acts on the *CDS* at least, else *EPS* deals with *CD* or *ADS*.

4) *SS* includes the states and their transformation. $SS = \{(S_i, T_i)\}$, where $S_i = \{s_{is}, \ldots, s_{ij}, \ldots, s_{ie}\}$, s_{ij} is the middle state in the S_i. $\forall S_i$, there exists a start state called s_s and a end state named as s_e; $T_i = \{ t_{ij} \mid t_{ij} = (s_{im}, s_{in}, p), op_k \in p$, *p* is a ordered set of op_k, $p \subset OPS$. t_{ij} represents that *BO* is transformed from the present state s_{im} to the next state $s_{in}\}$.

BO is classified into different categories according to their business functions in the certain industry, for example, the class of *Purchase Requirement*, the class of *Equipment Maintenance*, the class of *Warehouse Entry* etc. Every category of BO is an abstract set of business documents, which have the similar features. One of the classes is further divided into children classes according to the concrete semantics. For instance, the class of *Purchase Requirement* has children classes as follows: the requirement for production material, the requirement for equipment, the requirement for office etc.

BO describes all of the semantics at the range of business document. The semantics is partitioned into some semantic dimensions. Every semantic dimension contains three planes: data, operations and states. Every plane is made up of two axes. The data plane is $P(x, z)$, operations plane is $P(x, y)$, and states plane is $P(y, z)$. Thus, *semantic dimension* is defined as *BSD* (*id*, *BOID.ds*, *BOID.op*, *BOID.ss*), where $BOID.ds \subset DS$, $BOID.op \subset OPS$, $BOID.ss \subset SS$. When the operation is triggered in one dimension, the values of data and states will be changed in the same semantic dimension. **Figure 4** gives an example to illuminate what is semantic dimension in BO. The main semantic dimension of *Purchase Requirement* is a cross section, which is composed of *CDS*, *GPS* and *Running state*, where *CDS* = {*Bill identifier*, *Material item*, *Requirement amount*, *Requirement date*, …}, *GPS* = {*Insert*, *Modify*, *Delete*, *Release*, *Perform*, …} and *Running state* is *set* (S_{00}, T_{00}) = {{*NewCreated state*, *Released state*, *Running state*, *Finished state*}, {(*NewCreated state*, *Released state*, *Release*), (*Released state*, *Running state*, *Perform*)…}}.

We proceed to define the three relationships based on the semantics partition, which are shown in **Figure 2**. The definition of approval relationship is as follows:

Definition 2: ***Approval Relationship*** is defined as a set of *AP*, is a series of the approval actions. The approval action is represented as a node. The *AP* = (*StepID*, *Precondition*, *Roles*, *Operation*, *Next StepID*), where *StepID* is the node identifier. This node is visited when the precondition has been prepared. *Roles* = {*role* | *role has the rights to perform the approval operation*}. *Operation* = { *a* | *a represents op* $\wedge$ (*a = accept or a = reject*)}, *Next StepID* = { *b* | *b represents StepID*, where *b = x iff Operation = accept*, *or b = y iff Operation = reject*, *x*, *y* $\in$ {*StepID*}}.

After the auditing people finish the operations, the values of auditing dimension will be updated, and than the BO will be transformed to the next steps, in where the transformation path are predefined in *AP*. There exist many different approval paths for a BO, because the enterprises apply the extensive or intensive management pattern.

The auditing relationship is an ordered route in the auditing dimension. For instance, the approval dimension involves the following elements: ADS_i = {*approval action ID*, *people name for approval*, *approval date*, *advice*}, EPS_i = {*Query*, *Accept*, *Reject*} in the approval $action_m$, where $m=1,2,3,\ldots,n$. (S_{ij}, T_{ij}) = {{*Waiting State*, *Accepted State*, *Rejected State*}, {(*Waiting State for Approval*, *Approving*, *Accept in the action*$_1$),…, (*Approving*, *Accepted State*, *Accept in the action*$_n$)}}.

The approval relationship has effects on the inner of one BO. On the contrary, the association and integration relationships are built on the corresponding semantic dimensions among Business Objects (BOs). When two BOs are connected with the relationships of association or integration, the any operation of BO may change the state of the other BO.

For the association relationship between two BOs, a middle BO named as *Association Object* bridges many-to-many relationship of data. In **Figure 4**, BO3 is an association object between BO1 and BO2. The association object is a virtual object for software designer, is not an entity which is transacted by the business people.

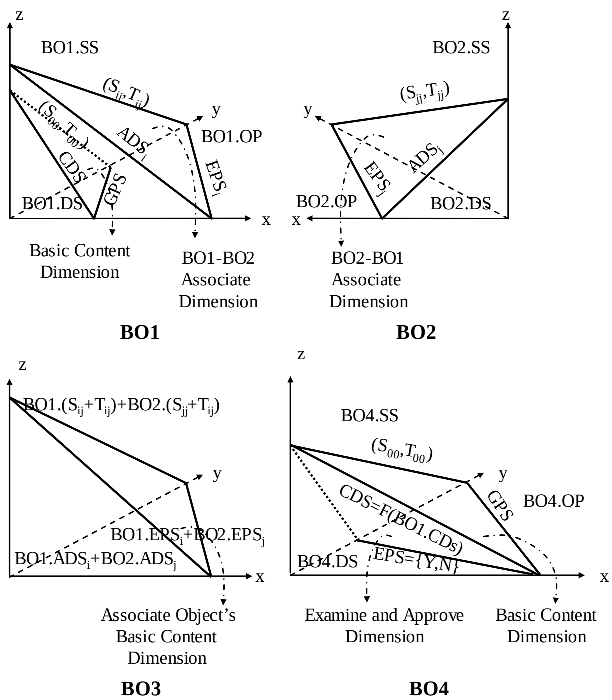


Figure 4. Business semantic dimensions.

Definition 3: ***Association Relationship***.

$\forall BO_1$, $BO_2 \in BO$, if BO_1 is associated to BO_2, the following condition must be prepared:

1) There is BO_2-oriented semantic dimension named as BSD_1 ($BO_1.ASS$, $BO_1.ADS_i$, $BO_1.EPS_i$, $BO_1.(S_i,T_i)$). At this moment, the every element of this tuple has special semantics. The data element of $BO_1.ADS_i$ is associated to the data element of BO_2. The EPS_i is an operation set, which connects the data mapping from BO_1 to BO_2. $BO_1.(S_i,T_i)$ represents the association states in this dimension of BO_1, whose values will be changed because of this mapping.

2) BO_2 also has the BO_1-oriented semantic dimension, named as BSD_2 ($BO_2.ASS$, $BO_2.ADS_j$, $BO_2.EPS_j$, $BO_2.(S_j,T_j)$), the others are similar to (1).

3) There exists one association object, represented as BO_3 (BO_3, $BO_3.DS$, $BO_3.OPS$, $BO_3.SS$), Where $BO_3.CDS = BO_1.ADS_i \cup BO_2.ADS_j$; $BO_3.GPS = BO_1.EPS_i \cup BO_2.EPS_j$; $BO_3.SS = \{(S_0, T_0), BO_1.(S_i, T_i), BO_2.(S_j, T_j)\}$, it is composed of the *Running States* (S_0,T_0) in BO_3 and association states.

The core data set of association object is composed of data in these BOs which associate with each other. There are three types of associations according to the functions of data items, such as the association between data items, the association between keys, the association between keys and attributions. In association object, the *ADS* contains data elements as follows: *Creator*, *Creating date*, *the Last Modifier*, *the Last Modifying Date* etc. In the complex system, the association is bidirectional and transmissible. For example, the Process 1 *in Figure 1* shows that there is a direct association between the *Purchase Requirement* and *Purchase Plan*. The numerous records of materiel requirement are combined into one purchase task of *Purchase Plan*. And then the *Purchase Order* is transformed from the *Purchase Plan*. Thus, the indirect association between the *Requirement* and *Order* is connected. The object of *Purchase Requirement* can build many direct or indirect association paths with the other objects in its lifecycle. The subsequent records in the business process can be traced to the sources along the paths. By doing so, the intensive management keeps at the fine granularity in the enterprise.

Definition 4: ***Integration Relationship.***

$\forall BO_1$, $BO_2 \in BO$, there exists BO_1(BO_1, $BO_1.DS$, $BO_1.OPS$, $BO_1.SS$) and BO_2(BO_2, $BO_2.DS$, $BO_2.OPS$, $BO_2.SS$). There is not less than one $d = F(BO_1.d_1,\ldots, BO_1.d_n)$ in $BO_2.CDS$, where F is formula used for computing value of BO_2. Thus, there exists the integration from BO_1 to BO_2.

In real enterprise, the part values of business object are obtained from many business objects through the complex computation. Therefore, the definition 4 can be extended. If there are integration relationships in the BO_1, BO_2, …, BO_n and BO_j, BO_j (BO_j, $BO_j.DS$, $BO_j.OPS$, $BO_j.SS$) must satisfy that there exists no less than one data item $d = F(BO_1.d_1, \ldots, BO_i.d_k, \ldots, BO_n.d_n)$, where F is formula used for the computation from BO_i to BO_j, $1 < i < n < j$. Generally, the formula F is various in different enterprises. Therefore, the implementation of F is second development by the programmers. The operation patterns of integration are classified into *Push Integration* and *Pull Integration*. The pattern called *Push Integration* need satisfy the following constraints: 1) the lifecycle of BO_1 is earlier than BO_2; 2) the operator in BO_1 is used for triggering the integration; 3) the BO_1 starts the integration, and BO_2 receives the messages. On the contrary, the *Pull Integration* is that the trigger operation belongs to BO_2, and BO_2 is defined as active object.

4. BOSD

4.1. Basic Thinking

In this section, we propose the software development approach that combines business object and business process together. The basic thinking is shown in **Figure 5**.

In the phase of requirement analysis, the consultators firstly analyze the management patterns and further build the models through researching the actuality of enterprise. First, the decision model is planar model which is an extended GRAI-GRID model. The decision model is

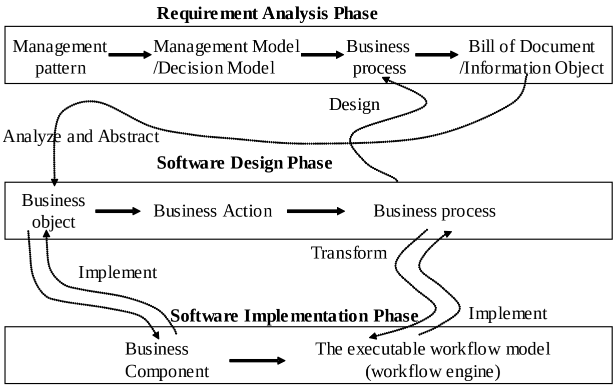


Figure 5. The process of BOSD.

defined as *Decision Model* = (*H*/*P*, *FUNC*, *ORDER*, *DecisionCenter*), where, *H/P* is row and *FUNC* is columns; *H/P* represents the different decision granularity, and is used for defining the time span (such as year or month) or different decision layers (such as stratagem layer, tactics layer and operations layer); A business target is generally realized by the collaborative activities, each of which is executed respectively by the different departments. *FUNC* is a set of functions. *FUNC* is described as a set of relationships, which are used for realizing a common target. The *DecisionCenter* is a cross cell of the row and column, and is defined as $\{order \mid order = f(h/p, func),\ h/p \in H/P,\ func \in FUNC\}$. The *order* represents an instruction, which is always saved in the business document. Second, we can produce the business process models through tracing the physical transformation from the materials to the products. The business process in the requirement analysis phase is defined as follows: *Process Model* = (*IP*, *ORG*, *DOC*, *ORDER*, *ACTION*'), where the IP is the information entity, represents an instruction or document in process model. The $IP = \{ip \mid ip = order \text{ or } ip = doc \text{ and } order \in ORDER,\ doc \in DOC\}$. The *ORG* represents the department set. *DOC* is a set of documents. The *ACTION'* defines the set of activity, which is executed by the *ORG* and acts on the *ORDER*. Finally, the information objects are abstracted from business documents which are used in the process models. Thus, the function-oriented requirements are collected clearly.

Aiming to enhance the independence of business object, the functions acting on the information objects are combined with the information objects to business object in the phase of software design. The business objects are connected to business processes by the business actions. For designing the flexible software using this approach, the approval, association and integration relationships are implemented by the particular approach in **Figure 6**. The software development is followed these steps: 1) the business component corresponding to the special BO is developed independently; 2) the inner relationships, such as the approval process and state transformation, are firstly considered to implementation; 3) when the whole business object is designed clearly, we connect them using the association relationship. And we define the state transformation which is affected by the associations under the control of association mechanism; 4) at last, the complicated integration relationship is developed by the extended mechanism, including automatic data import, computation and carrying forward etc.

In the development phase, business components are transformed from BOs, the relationships between them are implemented to executable workflow model. Thus, the workflow model is defined as (*BO.DS*, *BO.AT*, *CON*, *ATRelation*, *RoleSet*), where the *BO.DS* is the set of data; *BO.AT* represents the activity set, is mapping to the *BO.OPS*. $CON = \{condition_{x \to y} \mid condition \text{ is the transformation condition from the activity } x \text{ to } y\}$; The *ATRelation* is the transformation rules, is defined as $ATRelation = \{ f \mid y = f(x) \text{ iff } condition_{x \to y} = true,\ x \in BOi.AT,\ y \in BOj.AT,\ i,j = 1,\ldots,n\}$; *RoleSet* is the roles which take part in the *BO.AT*.

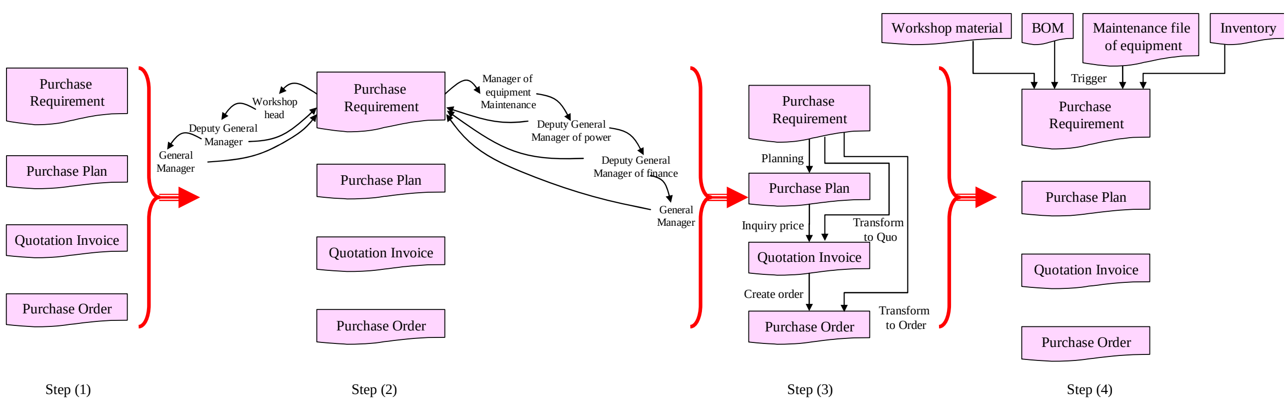


Figure 6. The steps of building the relationships between BOs.

4.2. Development Framework

The methodology introduced in Section 4.1 is very enormous and need to be supported by software technology. Hence, we choose the platform named as ICEMDA (Interoperable Configurable Executable Model Driven Architecture) [22,23] to implement this thinking. The platform has been developed mostly and is shown in **Figure 7**. There are five phases using this platform to develop the software for enterprise as follows: building the CIM for business requirement, building the PIM for the software design, and building the PSM for implementation on particular platform, generating the components from PSM, and assembling the components to a system.

The models in the requirement analysis phase are structured as decision model, business process model, information model and organization model on the support of using graphical modeling tools [24] on the CIM layer. The models on the CIM level are semi automatically transformed to the models on the PIM level, which involves business object models, BO-based workflow model, role-dependent model and data model. The three relationships above mentioned are described detailedly in the BO-based workflow model on the PIM layer. And then the PIM is refined to the PSM on J2EE platform, such as business component model (the model for component implementation) based on a specific software pattern, the executable workflow model, and the deployment model which defines the assembly of business component and workflow model. The business component is produced from the Platform-specific component model by code generator [25], and then can be secondly developed for meeting the special requirements. In the system for enterprise, the workflow engine parses the workflow models, and then choreographs the components to supply the different services to the clients.

4.3. BO and Relationships Expression

We illustrate the expression of BO and the relationships using a purchase management system. The business objects of purchase system are *purchase requirement*, *quotation*, *purchase plan*, *purchase order*, *arrival notice* and *advice of settlement* etc.

According to the definition 1, the contents of BO are described by the graphical platform-independent models in **Figure 8**. There are four types of platform-independent models which correspond to the data, state and operation set in Definition 1, and they are the data diagram, state diagram and class diagram. The use case diagram is mapping to the role-dependent model, which represents the roles act on the business objects in the workflow model.

The relationships between BOs in the Section 3 are described by a special relationship object called BOR (Business Object Relationship). The BOR can be implemented into three instances: association object, auditing object and integration object. And the BOR is also made up of the rules from the views of the data, operations and states. We can define the BOR = $\{f_R(BO_i, BO_j) \mid f_R = (r_1 \wedge \ldots \wedge r_k), r_1, \ldots, r_k \in R\}$, $R = \{ r \mid r = f(xs_m, xs_n), xs_m \in BO_i.XS, xs_n \in BO_j.XS, 1 \le m \le |BO_i.XS|, 1 \le n \le |BO_j.XS|$, where $x = d \mid op \mid s, X = D|OP|S$ }. The Process 4 in Figure 1 is modeling to the class diagram, which is shown in Figure 9. The bold lines represent the source process, and every BO has its own auditing object, just like the Auditing 2 for materiel. The graphical model is used for understanding easily, and further the model is

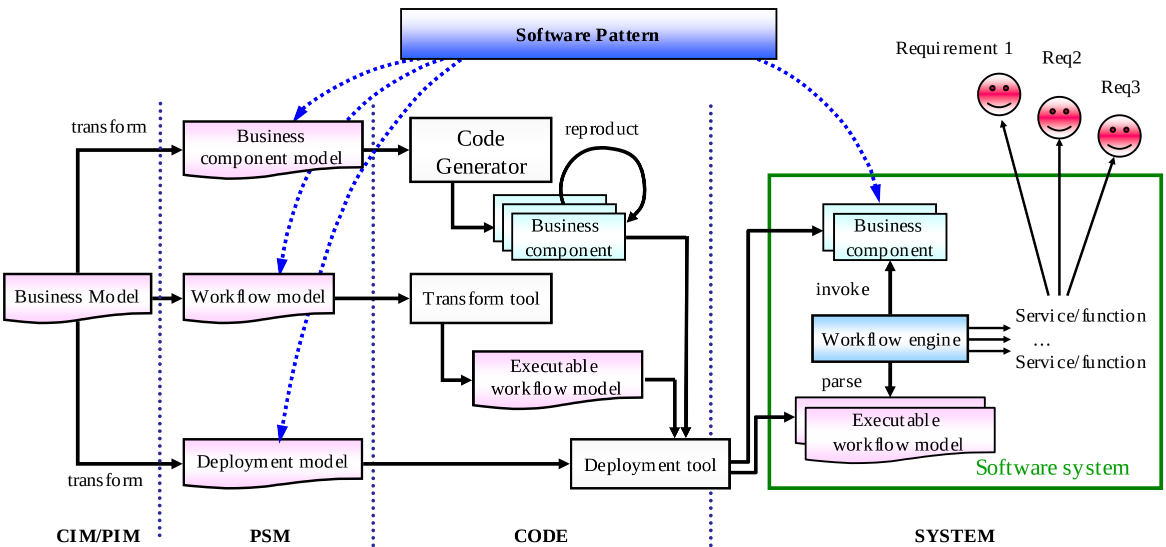


Figure 7. The ICEMDA framework.

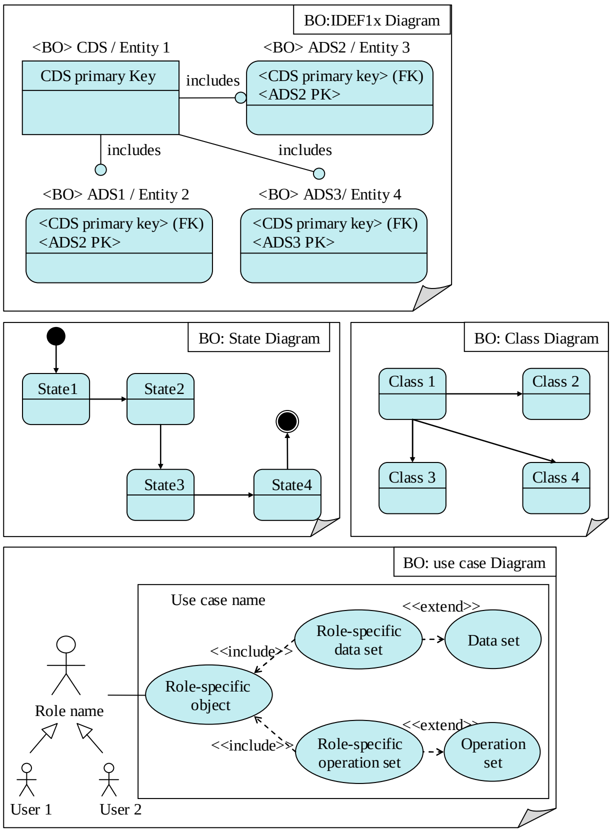


Figure 8. The BO platform-independent model.

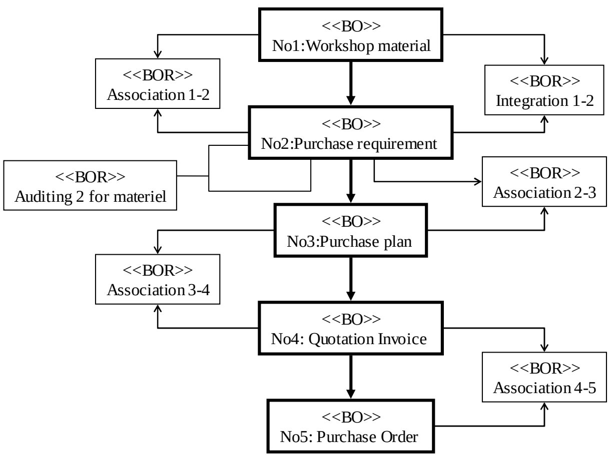


Figure 9. The class diagram of BOs.

saves as the file in XML (eXtensible Markup Language). Thus, the *BOR* is expressed as the fragment of XML.

When the *Purchase Requirement* is arranged into *Purchase Plan*, the association relationship between them has been built. It's defined as the following XML list:

<associationSet>
<ass AssBOId="No2-No3">
<AssedObject BOId="No2"
name="Requirement"
assString="Need_code,Material_code, Material_name,needNum">
</AssedObject>
<AObject BOId=" No3"
name="Purchase Plan"
assString="Plan_code,Material_ID,PlanNum">
</AObject>
<assCondition>
Material_code = Material_ID
</assCondition>
</ass>
</associationSet>

The second example is that *Purchase Order* is approved by the appropriate roles, each of which has the permission right according to the material type or money amounts. Thus the approval relationship of *Order* can be described as:

<ApprovalPath BOId="No5">
<step StepId="start"
personString="dept manager"
condition="money<=50000 and money>0 or material belongto 'steel'"
nextStep="accept?first:goback"></step>
<step StepId="first"
personString="assistant manager"
condition="money<=100000
and money>0"
nextStep="accept?second:goback">
</step>
<step StepId="second" ...></step>
<step StepId="end" personString="manager"
condition="money>100000"
nextStep="accept?gonext:goback"></step>
</ApprovalPath>

The third example: when the material in workshop lacks abruptly, the record of *Requirement No2* is achieved through importing the data of *Workshop Material Requirement.* For supporting this process, that integration relationship between these objects need be predefined. When the delivery date is satisfied, the integration formula F = {*Order*.*needNum* = ∑ *Requirement*. *Material-Num*}. The operator used for triggering the *Pull Integration* is defined as follows:

<extendOpSet>
<extendOp opID="ExtendedOP"
opName="import from Workshop Material Requirement ">
<whereRun BOId="Order"></whereRun>
<! —Pull Integration-->
<function> Order.needNum = sum (*Requirement*. MaterialNum)
</function>

<condition> Order. needNum, Order. time </condition>
…
</extendOp>
</extendOpSet>

5. Related Works

Several researchers have done work on developing enterprise software based on business object. Typical applications are requirement engineering, software design and implementation. Generally, the business object is used for implementation of business logic, which is composed with appropriate presentation object or web application to become an integrative application for the terminal clients. However, with the improvement of the enterprise software, business object is considered important object type. The business object type, technical object type and application object type are three types of objects based on the principle of separation of concern [26]. And then the business objects are specialized to common business objects, industry specific business objects, company specific business objects and user specific business objects according to the individual requirements of the different roles [27]. Therefore, the business objects are defined as components of the enterprise software that directly represent the business model. The largest granularity of business objects are cooperative business object (CBO) [28]. The CBO is considered the end product of the development process and cooperates with other objects to perform some desired task.

The different business objects have different application scenarios, therefore the characteristics of business object are diversity. In **Table 1** we give the comparisons between our business object and the common BO from OMG and the CBO from O Sims [28]. On one hand, the BO in this paper is lower level than the OMG's BO. On the other hand, the BO in this paper, which is the composition of many fine-grained objects, is one kind of the smaller scale CBO. These fine-grained objects always cooperate partially to meet the requirement in the range of business documents. The software development based on the different BOs is also comparatively diversity. We compared our BOSD to the business process-based methodology which is proposed by Somjit and Dentcho [14] under the background of **Figure 1**. The number of BOs in **Figure 1** is twenty, including eight BOs, eight auditing objects, four association objects and four integration BORs; the number of BO is eight in the method proposed by Somjit. Although they are coarse-grained objects, the reconfiguration cost of BOSD is mostly half of the other method. We only modify the information of the BORs to meet the processes changes.

According to the comparison above mention, our business objects are closest to the component-based implementation for enterprise software. A benefit of our approach is that it defines clearly the relationships between the business objects. The model for software design which is composed of business objects and the relationships is easily transformed on the MDA platform.

6. Conclusions

In this paper, aiming at the problem that current flexible software development methods lack of the systemic methodology and technology support, we present an approach based on coarse-grained business object. By analyzing the changeability of business processes and stability of business objects, we abstract an independent business object as the unit of development and reconfiguration. The three relationships among business objects are defined to describe the variable business processes. Thus, the business objects are assembled to system through their relationships. The implementation of this approach

Table 1. Comparisons between the BOs in literatures.

	OMG's BO	BOSD's BO	Sims's CBO
Counterpart in real enterprise	concept	Business document	The integration application for the user
Software lifecycle	All phases	design phase	All phases
Abstract degree	High	Low	Low
Architecture	undefined	defined	open
Independence	—	Y	Y
Granularity	—	large	larger
Reusable	Y	Y	Y
Easy to program	—	N	N

is supported by the ICEMDA platform.

In conclusion, there are several innovations in the method presented in this paper, as follows: 1) analyze the variable features of the enterprise system, and find out the flexible software can be composed of the changeable business processes and stabile business objects; 2) present an explicit definition of coarse-grained business object; 3) make clear the typical relationships between the business objects; 4) present an method for the flexible software development based on the stable business object, which is implemented by the MDA.

Future works includes: further research on the other relationships between business objects, automatic identification from the business processes, complete the automatic transformation from the PIM to PSM on the platform ICEMDA, etc.

5. Acknowledgement

This work is supported by State 863 High-Tech Program (No. 2009AA04Z153) and State Natural Science Foundation (No. 60773064) of China. The author would like to thank reviewer's helpful suggestions for revising this paper.

REFERENCES

[1] J. Kramer and J. Magee, "The Evolving Philosophers Problem: Dynamic Change Management," *IEEE Transactions on software engineering*,Vol. 16, No.11, 1990, pp. 1293-1306.

[2] W. J. Kettinger and V. Grover, "Special Section: Toward a Theory of Business Process Change Management," *Journal of Management Information Systems*, Vol. 12, No.1, 1995, pp. 9-30.

[3] D. Kim, M. Kim and H. Kim, "Dynamic Business Process Management Based on Process Change Patterns," *Proceedings of the* 2007 *International Conference on Convergence Information Technology*, Gyeongju, 2007, pp. 1154- 1161.

[4] P. Soffer, B. Golany and D. Dori, "Aligning an ERP System with Enterprise Requirements: An Object-Process Based Approach," *Computers in Industry*, Vol. 56, No.6, 2005, pp. 639-662.

[5] G. Pour, "Moving toward Component-Based Software Development Approach," *Proceedings of the* 27*th Technology of Object-Oriented Languages* (*TOOLS* 27), Beijing, China, 1998, pp. 296-300.

[6] J. Kotlarsky, I. Oshri, K. Kumar and J. van Hillegersberg, "Towards Agility in Design in Global Component-Based Development," *Communications of the ACM*, Vol. 51, No. 9, 2008, pp. 123-127.

[7] Architecture Board ORMSC. Model Driven Architecture (MDA). OMG document number ormsc/2001-07-01. http://www.omg.org/cgibin/doc?ormsc/2001-07-01.pdf, (accessed 1/7/2001)

[8] E. Breton and J. Bézivin, "Model Driven Process Engineering," *Proceedings of the* 25*th Annual International Computer Software and Applications Conference* (*COMPSAC* 25*th*), Chicago, Illiois, 2001, pp. 225-230.

[9] J. Koehler, R. Hauser, S. Kapoor, F. Y. Wu and S. Kumaran, "A Model-Driven Transformation Method," *Proceedings of the Seventh Enterprise Distributed Object Computing Conference*, Brisbane, Queensland Australia, 16-19 September 2003, pp. 186- 197.

[10] M. P. Gervais, "Towards an MDA-Oriented Methodology," *Proceedings of the* 26*th Annual International Computer Software and Applications Conference* (*COMPSAC* 2002), England, 26-29 August 2003, pp. 265-270.

[11] M. Keith, H. Demirkan and M. Goul, "Service-Oriented Software Development," *Proceedings of the* 2009 *Americas Conference on Information Systems* (*AMCIS* 2009), 2009, America, p. 100.

[12] P. Wohed, W. Vander Aalst, M. Dumas and A. Ter, "On the Suitability of BPMN for Business Process Modeling," *Business Process Management*, Vol. 4102, 2006, pp. 161-176.

[13] D. Hollingsworth, "The Workflow Reference Model," *Technical report, Workflow Management Coalition*, TC00-1003, December 1994.

[14] A. Somjit and B. Dentcho, "Development of Industrial Information Systems on the Web Using Business Components," *Computers in Industry*, Vol. 50, No. 2, 2003, pp. 231-250.

[15] S. Arbaoui, J. C. Derniame, F. Oquendo and H. Verjus, "A Comparative Review of Process-Centered Software Engineering Environments," *Annals of Software Engineering*, Vol. 14, No. 1-4, 2002, pp. 311-340.

[16] R. Sangwan, C. Neill, M. Bass and Z. E. I. Houda, "Integrating a Software Architecture-Centric Method into Object-Oriented Analysis and Design," *Journal of Systems and Software*, Vol. 81, No. 5, 2008, pp. 727-746.

[17] I. Reinhartz-Berger, D. Dori and S. Katz, "OPM/ Web-Object-Process Methodology for Developing Web Applications," *Annals of Software Engineering*, Vol. 13, No. 1-4, 2002, pp.141-161.

[18] H. Liu and D. P. Gluch, "Conceptual Modeling with the Object-Process Methodology in Software Architecture," *Journal of Computing Sciences in Colleges*, Vol. 19, No. 3, 2004, pp.10-21.

[19] W. Kozaczynski, "Architecture Framework for Business Component," *Proceedings of the Fifth International Conference on Software Reuse*, Canada, 2-5 June 1998, pp. 300-307.

[20] A. Karakaxas, B. Karakostas and V. Zografos, "A Business Object Oriented Layered Enterprise Architecture," *Proceedings of the* 11*th International Workshop on Database and Expert Systems Applications*, Greenwich, London, U.K., 4-8 September 2000, pp. 807-810

[21] M. Tsagias and B. Kitchenham, "An Evaluation of the Business Object Approach to Software Development," *Journal of Systems and Software*, Vol. 52, No. 2-3, 2000, pp. 149-156.

[22] L. Nie, X. Xu, C. David, Z. Gregory and D. Zhan, "GRAI-ICE Model Driven Interoperability Architecture for Developing Interoperable ESA," *The international conference on Interoperability for Enterprise Software and Applications*, United Kingdom, 12-15 April 2010, pp. 111-121.

[23] D. Zhan, J. Feng, L. Nie and X. Xu, "ICEMDA: An Interoperable Configurable Executable Model Driven Architecture," *Chinese Journal of Electronics*, Vol. 36, No. 12A, 2008, pp. 120-127.

[24] J. Li, D. Zhan, L. Nie and X. Xu, "Design and Implementation of a MOF Based Enterprise Modeling Tool," I-ESA, China, 2009.

[25] J. Feng, D. Zhan, L. Nie and X. Xu, "Pattern based Code Generation Method for the Business Component," *Chinese Journal of Electronics*, Vol. 36, No. 12A, 2008, pp. 19-24.

[26] R. Shelton, "Business Objects Response from Open Engineering Inc. to OMG BODTF RFI-1," http://www.OMG.com/OMG Document bom/97-10-07

[27] C. Casanave, "Business-Object Architectures and Standards," *Proceedings of the* 10*th Annual Conference on Object-Oriented Programming Systems, Languages, and Applications* (*OOPSLA'*95), USA, 1995.

[28] O. Sims, "Business Objects: Delivering Cooperative Objects for Client-Server," *The IBM McGraw-Hill Series McGraw-Hill Book Company*, 1994.

J. Software Engineering & Applications, 2010, 3, 926-932
doi:10.4236/jsea.2010.310109 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Investigation of Noise-Resolution Tradeoff for Digital Radiographic Imaging: A Simulation Study

Eri Matsuyama[1], Du-Yih Tsai[1], Yongbum Lee[1], Katsuyuki Kojima[2]

[1]Department of Radiological Technology, Graduate School of Health Sciences, Niigata University, Niigata, Japan; [2]Department of Business Administration, Graduate School of Business Administration, Hamamatsu University, Hamamatsu, Japan.
Email: meri@cyber.ocn.ne.jp, {tsai, lee}@clg.niigata-u.ac.jp, kojima@hamamatsu-u.ac.jp

Received July 16th, 2010; revised August 9th, 2010; accepted August 16th, 2010.

ABSTRACT

In digital radiographic systems, a tradeoff exists between image resolution (or blur) and noise characteristics. An imaging system may only be superior in one image quality characteristic while being inferior to another in the other characteristic. In this work, a computer simulation model is presented that is to use mutual-information (MI) metric to examine tradeoff behavior between resolution and noise. MI is used to express the amount of information that an output image contains about an input object. The basic idea is that when the amount of the uncertainty associated with an object before and after imaging is reduced, the difference of the uncertainty is equal to the value of MI. The more the MI value provides, the better the image quality is. The simulation model calculated MI as a function of signal-to-noise ratio and that of resolution for two image contrast levels. Our simulation results demonstrated that MI associated with overall image quality is much more sensitive to noise compared to blur, although tradeoff relationship between noise and blur exists. However, we found that overall image quality is primarily determined by image blur at very low noise levels.

Keywords**:** *Modeling and Simulation, Medical Imaging, Image Quality Evaluation, Mutual Information*

1. Introduction

The modulation transfer function (MTF), noise power spectrum (NPS), and detective quantum efficiency are commonly used as image quality metrics to characterize resolution, noise, and efficiency performance of digital radiographic systems, respectively [1-3]. These metrics are dealt with in the spatial frequency domain. Recently, an information-entropy based approach has been reported for evaluating overall image quality in medical imaging systems [4-6]. In these reports, transmitted information (TI) [4,5] or mutual information (MI) [6] was used as an image quality criterion. Both TI and MI were defined as "the amount of shared information", *i.e.*, "the amount of information transmitted from stimulus (input) to response (output)". The more the transmitted information provides, the better the image quality is. Therefore, the overall quality of an image can be quantitatively evaluated by measuring TI (or MI). Unlike the physical performance measures, the information entropy-based metric is dealt with in the spatial domain.

One of the current dilemmas in digital radiography is the extent to which these parameters such as, resolution and noise affect physical or clinical image quality. An imaging system may only be superior in one metric while being inferior to another in the other metric. In general, higher spatial resolution leads to an increased noise level. Simulation studies of image quality attributes for x-ray systems using computer methods have been performed by several investigators, and shown to be effective methods of evaluating various elements of the image formation process [7,8]. A computer simulation approach was also presented to investigate the impact of image quality metrics on the appearance of radiographic images [9]. The approach was to emulate the influence of resolution and noise characteristics of a digital detector on the appearance of a radiographic image. Recently, attention has been paid to address the tradeoff between spatial resolution and quantum noise relation for computed tomography and digital radiography [10-12]. In these studies, it is of general nature that the MTF and NPS were used as the descriptors of spatial resolution and noise. However, we believe that it is also interesting to attempt to understand the tradeoff in terms of image information such as mutual information.

In this paper, a computer simulation approach is presented that is to employ the MI metric to investigate tradeoff behavior between resolution and noise. The simulation model calculated MI as a function of signal-to-noise ratio and that of resolution for two specific image contrast levels. Two simulation studies were per- formed separately; the first simulation was carried out to investigate the relationship between image blurring and MI for various levels of noise, and the second simulation was conducted to investigate the relationship between image noise and MI for different extent of blurring. In this work, a total of 2,688 simulations were performed in order to conduct a detailed analysis and achieve a better understanding of noise-resolution tradeoff.

2. Theoretical Framework

MI is a concept from information theory [13,14] and is also referred to as TI as described in the section of preceding Section [4-6]. MI has been applied in medical image processing, particularly for image registration [15-18]. The definition of the term of MI has been presented in various ways in the literature. We will briefly describe MI, as used in the image evaluation sense, rather than as used in the image registration sense.

Given events $s_1, \ldots .. s_n$ occurring with probabilities p_1, $p_2, \ldots \ldots p_n$, the Shannon entropy H is defined as

$$H(p_1, p_2, \ldots p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i \quad (1)$$

Considering x and y as two random variables corresponding to an input variable and an output variable, the entropy for the input and that for the output are denoted as $H(x)$ and $H(y)$, respectively. For this case, the MI can be defined as

$$\begin{aligned} MI(x;y) &= H(x) - H_y(x) = H(y) - H_x(y) \\ &= H(x) + H(y) - H(x,y) \end{aligned} \quad (2)$$

where $H(x,y)$ is the joint entropy, and $H_x(y)$ and $H_y(x)$ are conditional entropies. The entropies and joint entropy are given as

$$H(x) = -\sum_{i} p_i \log_2 p_i \quad (3)$$

$$H(y) = -\sum_{j} p_j \log_2 p_j \quad (4)$$

$$H(x,y) = -\sum_{ij} p_{ij} \log_2 p_{ij} \quad (5)$$

where p_i and p_j are the marginal probabilities, and p_{ij} is the joint probability.

A useful way of visualizing the relationship among these entropies is provided by a Venn diagram as shown in **Figure 1**. The MI measures how much the uncertainty of input x is known if output y has been given. It can be easily shown that if input and output are generally independent, then $H(x,y) = H(x) + H(y)$. Consequently, their MI is zero (*i.e.*, transmitted information is equal to zero). In other words, observing y does not reduce the uncertainty of x. If, however, $x = y$, *i.e.*, $H(x) = H(y)$, then MI = $H(x)$. Under this condition, the information about input x can be obtained completely. We apply the MI measure to evaluate image quality of digital radiography based on the following reasoning. Consider an experiment in which every input has a unique output belonging to one of the various output categories. The inputs may be considered to be a set of subjects, for example, a test sample object with steps of various thickness, whereas the outputs may be their corresponding images varying in optical density or gray level. If the inputs can be recognized completely when the outputs have shown, then the quality or the performance of the transmission channel of the system (*i.e.*, imaging system), can be perfect. In the cur- rent study, a method of occurrence-frequency-based computation was used for calculating the entropies of input, output, and their joint entropies. The details of the calculation procedure can be found in the literature [4-6].

3. Methods

3.1. Computer Simulation

A simulation was designed, and its framework is as follows. A simulation image $g(x,y)$ was given by a spatial convolution between a uniformly-distributed signal (an object) $f(x,y)$ having intrinsic noise $u(x,y)$ and a blurring function B. If the external noise $v(x,y)$ was also taken into consideration, the resulting image could be represented by the following formula:

$$\begin{aligned} g(x,y) = \sum_{s=1}^{5} \{ & [s \times f(x,y) + u(x,y) \times W] * B \\ & + v(x,y) \times K \}, \end{aligned} \quad (6)$$

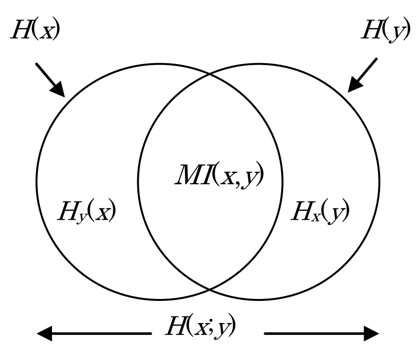


Figure 1. The relations between conditional and joint entropies, and the mutual information.

where the symbol $*$ represents the convolution operation, and s is an integer representing the number of steps of the simulated image. The terms of W and K are weighting coefficients used to adjust noise level.

In the simulation studies, the input image $f(x,y)$ was a five-step wedge with a specific intensity or a pixel value on each step. Both $u(x,y)$ and $v(x,y)$ were zero-mean Gaussian noise with a standard deviation of 0.5. In the studies, for simplicity, the term of $v(x,y) \times K$ was considered the external noise and is equal to the intrinsic noise of $u(x,y) \times W$ (*i.e.*, $u(x,y) \times W = v(x,y) \times K$). We used a "$m \times m$" ($m$ is an odd integer) 8-neighborhood averaging filter as the blurring function. The extent of blurring was adjusted by varying the filter size (FS). The reason for choosing neighborhood averaging filter was its ease of implementation and effectiveness, which were confirmed by experiments.

Two simulations were performed. The first simulation was carried out to investigate the relationship between image blur and MI for various noise levels at specific image contrasts levels. In this study, we defined image contrast as the difference of the mean pixel values between two adjacent steps of a simulated step wedge. We used signal-to-noise ratio (SNR) to describe the extent of noise level. Notice that the signal and noise used for SNR calculation were $f(x,y)$ and $u(x,y) \times W$, respectively, as given in Equation (6). In this work, combinations of 64 various SNRs (range, 24-43 dB), 21 various FSs (range, 3-41), and two different contrast levels (20 and 40) were used for simulation studies. As a result, a total of 2688 simulations were performed for the analysis of resolution-noise tradeoff.

The second simulation was performed to investigate the relationship between image noise and MI for different extent of blurring at specific image contrast levels.

3.2. Step-wedge Phantom Images

For verification and validation of our designed computer simulation models, phantom images of an acrylic step-wedge with 2, 4, 6, 8, and 10 mm in thickness were obtained using the following exposure conditions. The specific exposure factors were kept at 42 kV and 10 mA, the focus-imaging distance was taken at 185 cm, and the exposure time was varied ranging from 0.1 to 0.5 sec. An imaging plate for computed radiography was used as a detector to record x-ray intensities.

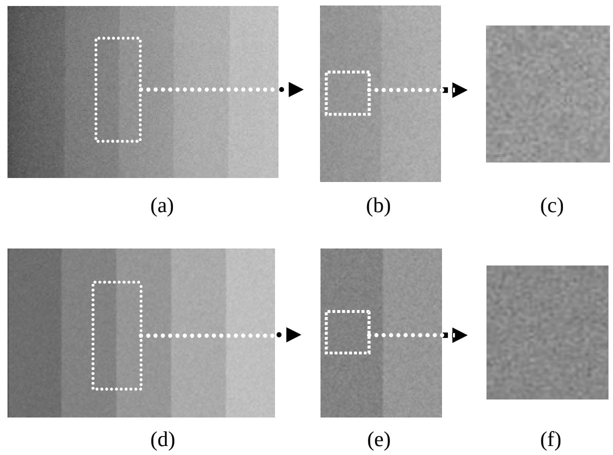


Figure 2. Perceptual evaluation of computer simulated images. (a) Computer-simulated image. The parameters used are: *W* = 130, SNR = 26.1 dB, contrast = 70, FS = 3. (b) The magnified image from the rectangular area shown in (a). (c) The magnified image from the rectangular area shown in (b). (d) Step wedge phantom image. Exposure time was 0.5 s. The step wedge was placed 30 cm apart from the center toward the cathode end for imaging. (e) The magnified image from the rectangular area shown in (d). (f) The magnified image from the rectangular area shown in (e).

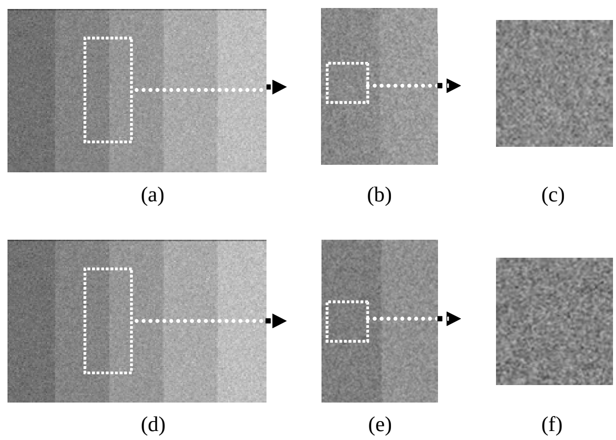


Figure 3. Perceptual evaluation of computer simulated images. (a) Computer-simulated image. The parameters used are: *W* = 300, SNR = 19.3 dB, contrast = 70, FS = 3. (b) The magnified image from the rectangular area shown in (a). (c) The magnified image from the rectangular area shown in (b). (d) Step wedge phantom image. Exposure time was 0.1 s. The step wedge was placed 30 cm apart from the center toward the cathode end for imaging. (e) The magnified image from the rectangular area shown in (d). (f) The magnified image from the rectangular area shown in (e).

4. Simulation Results and Discussion

Figures 2 and **3** compare the computer-simulated images versus the phantom images obtained. The simulated images shown in the figures were generated using Equation (6). A perceptual comparison of the simulated images and phantom images indicates that these images were very similar in appearance with respect to noise, blur and visibility of detail. The comparison result indicated that our designed mathematical model provides a good means of simulating the resolution and noise characteristics of digital radiographic systems.

Figure 4 illustrates MI as a function of FS for varying levels of SNR ranging from 24 to 41 dB at an image contrast of 40. It should be noted that FS is associated with the extent of blurring: the greater the FS value is, the higher the extent of blurring becomes. As shown in Figure **4**, MI decreases with the increase of FS (*i.e.*, increase of image blur) when noise levels are very low (*i.e.*, high SNR; for example, SNR>36 dB in this study), although the decrease is relative small. This means that, in the case of low noise levels, the effect of the level of blur on the MI is not so obvious in comparison to noise.

When noise increases to medium levels (for example, 36 dB ≥ SNR ≥ 31 dB in this report), MI initially increases with the increase of FS and then gradually decreases after reaching the maximum value. The increase in MI value might be because that, in spite of deterioration of image resolution, the increase of FS could give rise to a significant decrease of noise. Thus, MI is greatly dependent on the decrease of noise level compared to the deterioration of image resolution. However, on the contrary, when FS increases to a certain level, the MI value is greatly influenced by the deterioration of resolution as compared to that by the decrease of noise.

In the case of high noise levels (for example, SNR ≤ 28 dB), MI increases gradually with the increase of FS until reaching its maximum value. After that, MI value showed insignificant decreasing. The reasoning could be made as follows: 1) initially, the increase of FS might result in a great decrease of noise, and this yields the increase of MI, although the increase of FS itself might give rise to a small decrease of MI. In other words, the decrease of noise level dominated the variation of MI. 2) However, when FS continued increasing, a *tradeoff point* appeared. The location, indicated by an arrow on each graph shown in **Figure 4**, was referred to as the tradeoff point in this study. The location corresponds to the maximum value of MI. For instance, the tradeoff points for SNR = 41, 31, and 25 dB can be found at FS = 3, 9, and 17, respectively. It is noted that, not surprisingly, the location of the tradeoff point depends on SNR. An oblique line in the figure depicts the trend in the change of the location. As shown in the figure, MI reaches to its maximum at a finer resolution when SNR increases.

Figure 5 plots MI as a function of FS for varying levels of SNR at an image contrast of 20. Overall, the trend of this case is similar to that at image contrast of 40. It can be seen from **Figures 4** and **5** that images with higher contrast show greater MI values in comparison to those with lower contrast, if both images have the same spatial resolution (same FS) and the same noise level (same SNR). It is reasonable to conclude that a higher contrast image shows better image quality.

Figure 6 shows MI as a function of SNR for various

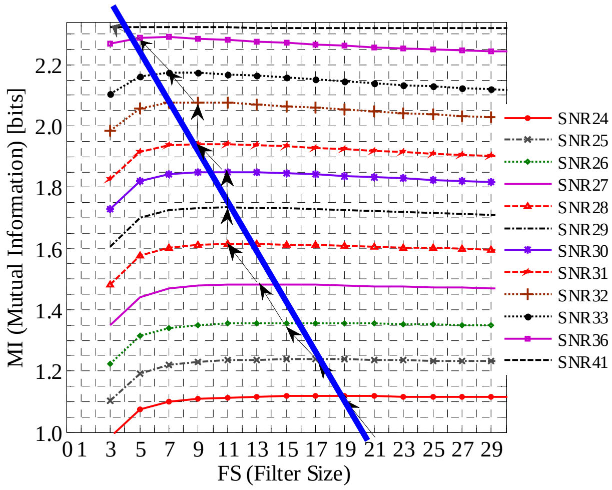


Figure 4. Relationship between FS and MI for varying levels of SNR at an image contrast of 40.

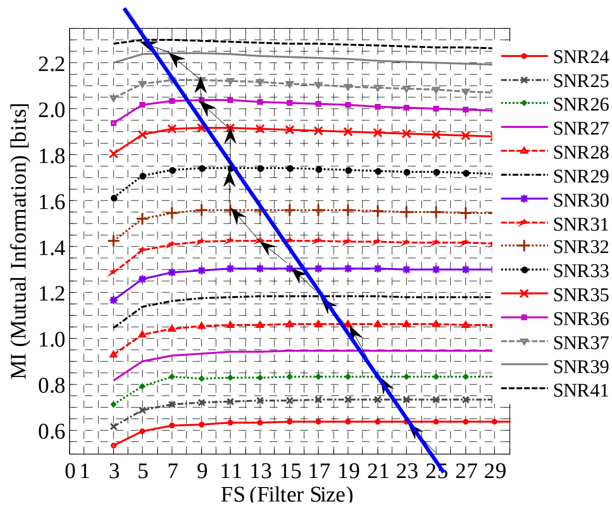


Figure 5. Relationship between FS and MI for varying levels of SNR at an image contrast of 20.

sizes of FS at an image contrast of 40. The results indicate that, basically, MI value increases with the increase of SNR (decrease in noise level). The figure illustrates only the results covering a range of SNR from 29 to 36 dB, where intersections between the line graphs occur. The intersection between two graphs indicates that two images have the same overall image quality, although the images may show different extent of blur. For instance, the intersection between the graphs of FS = 3 and FS = 41 is located near SNR = 33. It is noted that an image blurred by a smoothing filter of FS = 41 might decrease the MI value because of resolution deterioration caused by blurring. On the contrary, the MI value might increase because of the decrease of noise resulting from smoothing operation. This means that the physical quality of an image is mutually adjusted by resolution and noise properties at a specific image contrast. The result implies that the contribution of resolution attribute and that of noise attribute to the MI value vary depending on the levels of noise and blur. Here, it should be noted that the external noise (*i.e.*, the term of $v(x,y) \times K$ shown in equation (6)) also served as a factor that influences overall image quality of an image.

Figure 7 shows MI as a function of SNR for various sizes of FS at image contrast of 20. The figure also shows that the MI value increases with the increase of SNR. From **Figures 6** and **7**, it is noted that the MI value for the image with lower contrast is lower than that with higher contrast. However, the trends of the two cases are similar. As described earlier, a higher contrast image shows better image quality compared to a lower contrast image.

Figures 8 and **9** show the MI values, plotted as a function of SNR and FS (blur) at image contrast of 40 and 20, respectively. **Figure 8** corresponds to **Figures 4** and **6**, while **Figure 9** corresponds to **Figures 5** and **7**. As shown in the figures, MI reaches to the maximum value when an image has very high SNR. Moreover, it is noted that MI is much more sensitive to noise compared to blur. It is also noted that the image with a higher contrast level provides greater MI and shows better image quality in comparison with that with lower contrast level, if the two images have the same noise and blurring levels (**Figures 8** and **9**).

Figures 10 and **11** are plots of SNR (noise) versus FS (blur) for four different MI values (*i.e.*, 2.276, 2.206, 2.159, and 2.090), corresponding to four various transmitted efficiency (*i.e.*, 98%, 95%, 93%, and 90%) for the

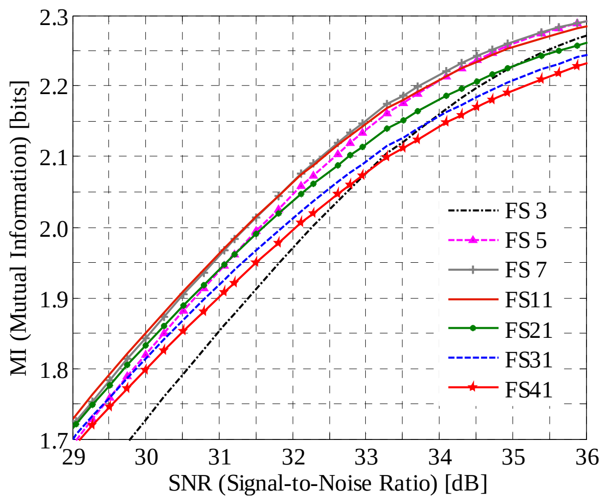


Figure 6. Relationship between SNR and MI for varying sizes of FS at an image contrast of 40.

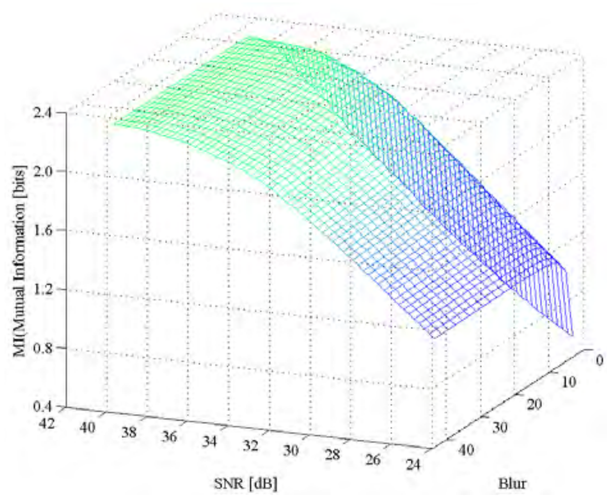


Figure 8. MI calculated as a function of SNR and FS (blur) at image contrast of 40.

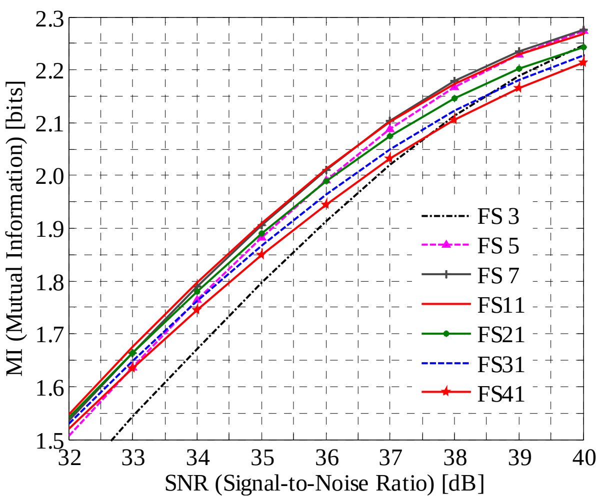


Figure 7. Relationship between SNR and MI for varying sizes of FS at an image contrast of 20.

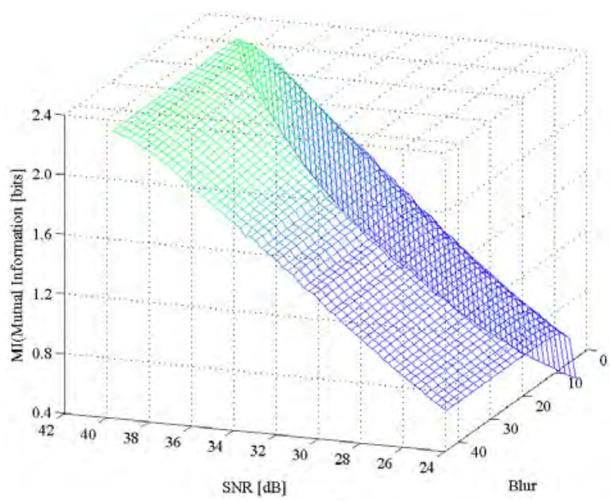


Figure 9. MI calculated as a function of SNR and FS (blur) at image contrast of 20.

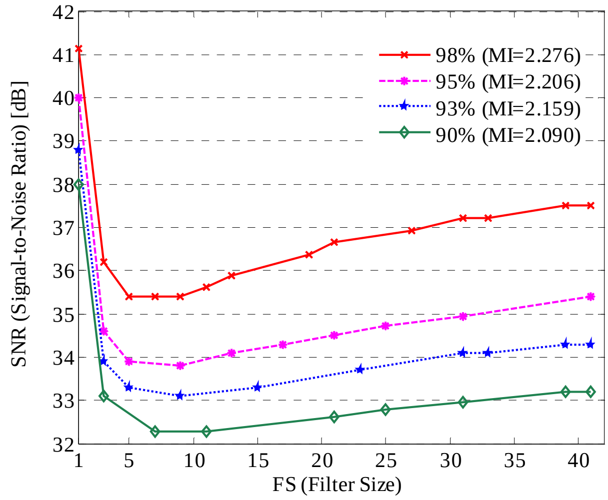


Figure 10. A plot of SNR versus FS for four different MI values at an image contrast of 40.

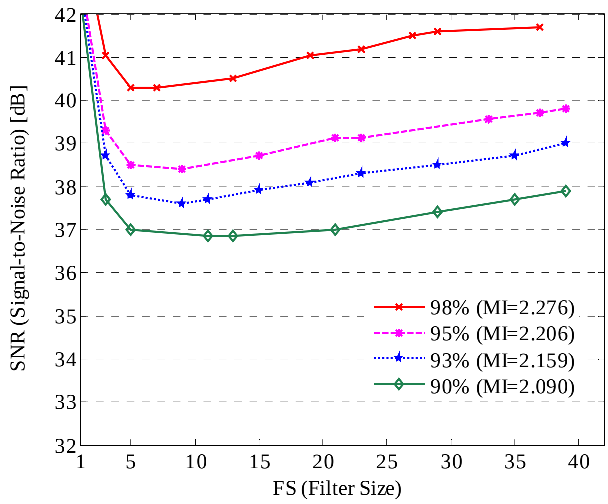


Figure 11. A plot of SNR versus FS for four different MI values at an image contrast of 20.

cases of contrast levels of 40 and 20, respectively. In this study, "transmission efficiency" was used and defined as the ratio between MI and the input entropy, *i.e.*, η=(MI/$H(x)$) %. The efficiency η may also be used as a measure for indicating how good the imaging quality of an image receptor is. Points shown on each curve in the figures, obtained from different combinations of SNR and FS, have the same MI values, thus providing the same overall image quality. It is observed that a minimum point exists at each graph. Noted that only high SNR levels (*i.e.*, low noise level) ranging from 32 to 42 dB are depicted in **Figures 10** and **11**. Because the minimum points for those SNR levels lower than 32 dB did not appear in our simulation studies. As shown in the two figures, tradeoff relationship between image noise and blur exists on the right of the minimum point. This means that a combination of a lower noise level and a deteriorated resolution might provide the same physical image quality as a combination of a higher noise level and a higher resolution level. On the left of the minimum point, however, image quality is primarily determined by resolution when the SNR of an image is higher than 32 dB in our investigation. In other words, the image quality of a very-low-noise image is almost determined by the extent of blur, even when noise level had slight variations. There might be two reasons for this. First, for images of very low noise levels, image quality might not be affected by small change of noise levels. As described in the section of Theoretical Framework, MI measures how much the uncertainty of input is known if output has been given. As a result, very small changes in the amount of noise might not influence the amount of the uncertainty. Second, as shown in **Figures 4** and **5**, MI has a significant increase with the increase of deterioration of resolution at low FS values (range, 3-9 in the simulation studies).

It must be addressed here that the purpose of this study was to present a computer simulation approach to investigate tradeoff behavior between resolution and noise using the MI metric. In order to validate the results obtained from this study, we will perform visual evaluation and compare them in the future work.

Our simulation results showed that the proposed simulation approach by employing mutual-information metric to examine tradeoff behavior between resolution and noise is useful, reliable and challenging.

5. Conclusions

A computer simulation study for examining resolution-noise tradeoff behavior has been presented. In this study MI was used as an image quality metric for the analysis. Our simulation results demonstrated that the MI value associated with overall image quality is much more sensitive to noise compared to blur, although tradeoff relation between noise and blur exists. However, at very low noise levels (SNR values higher than 32 dB), we found that overall image quality is primarily determined by image blur. However, a comparison between the result of physical evaluation and that of perceptual evaluation was not made in this work. It would be a very interesting research question for our future study.

REFERENCES

[1] H. Fujita, K. Doi and M. L. Giger, "Investigation of Basic Imaging Properties in Digital Radiography 6 MTFs of II-TV Digital Imaging Systems," *Medical Physics*, Vol. 12, No. 6, 1985, pp. 713-720.

[2] M. L. Giger, K. Doi and H. Fujita, "Investigation of Basic Imaging Properties in Digital Radiography 7 Noise Wiener Spectra of II-TV Digital Imaging Systems," *Medical Physics*, Vol. 13, No. 2, 1986, pp. 131-138.

[3] A. Maidment and M. Yaffe, "Analysis of Spatial-Frequency-Dependent DQE of Optically Coupled Digital Mammography Detectors," *Medical Physics*, Vol. 21, No. 6, 1994, pp. 721-729.

[4] E. Matsuyama, D. Y. Tsai, Y. Lee, M. Sekiya and K. Kojima, "Physical Characterization of Digital Radiological Images by Use of Transmitted Information Metric," *Proceedings of 2008 SPIE Medical Imaging*, Vol. 6913, 2008, pp. 69130V1-V8.

[5] D. Y. Tsai, Y. Lee and E. Matsuyama, "Information Entropy Measure for Evaluation of Image Quality," *Journal of Digital Imaging*, Vol. 21, No. 3, 2008, pp. 338-347.

[6] E. Matsuyama, D. Y. Tsai and Y. Lee, "Mutual Information-Based Evaluation of Image Quality with Its Preliminary Application to Assessment of Medical Imaging Systems," *Journal of Electronic Imaging*, Vol. 18, No. 3, 2009, pp. 033011-1-11.

[7] R. Nishikawa and S. Lee, "Preliminary Results on a Method for Producing Simulated Mammograms," *presented at the AAPM 44th Annual Meeting*, Montreal, Quebec, Canada, 2002.

[8] A. Tingberg, C. Herrmann, J. Besjakov, A. Almen, P. Sund, D. Adliene, S. Mattsson, L. Mansson and W. Panzer, "What is Worse: Decreased Spatial Resolution or Increased Noise," *Proceedings of SPIE Medical Imaging*, Vol. 4686, 2002, pp. 338-346.

[9] R. S. Saunders Jr. and E. Samei, "A Method for Modifying the Image Quality Parameters of Digital Radiographic Images," *Medical Physics*, Vol. 30, No. 11, 2003, pp. 3006-3017.

[10] T. Fuchs and W. Kalender, "On the Correlation of Pixel Noise, Spatial Resolution and Dose in Computed Tomography," *Physica Medica*, Vol. XIX, No. 2, 2003, pp. 153-164.

[11] B. Li, G. B. Avinash and J. Hsieh, "Resolution and Noise Trade-off Analysis for Volumetric CT," *Medical Physics*, Vol. 34, No. 10, 2007, pp. 3732-3738.

[12] A. R. Pineda and H. H. Barrett, "What Does DQE Say About Lesion Detectability in Digital Radiography," *Proceedings of 2001 SPIE Medical Imaging*, Vol. 4320, 2001, pp. 561-569.

[13] C. E. Shannon, "Mathematical Theory of Communication," *Bell System Technology Journal*, Vol. 27, 1948, pp. 379-423, pp. 623-656.

[14] C. E. Shannon and W. Weaver, "The Mathematical Theory of Communication," The University of Illinois Press, Urbana, 1948.

[15] D. Skerl, B. Likar and F. Pernus, "A Protocol for Evaluation of Similarity Measures for Rigid Registration," *IEEE Transactions on Medical Imaging*, Vol. 25, No. 6, 2006, pp. 779-791.

[16] G. D. Tourassi, B. Harrawood, S. Singh and J. Y. Lo, "Information-Theoretic CAD System in Mammography: Entropy-Based Indexing for Computational Efficiency and Robust Performance," *Medical Physics*, Vol. 34, No. 8, 2007, pp. 3193-3204.

[17] J. P. W. Pluim, J. B. A. Maintz amd M. A. Viergever, "Mutual-Information-Based Registration of Medical Images: A Survey," *IEEE Transactions on Medical Imaging*, Vol. 22, No. 8, 2003, pp. 986-1004.

[18] G. D. Tourassi, E. D. Frederick, M. K. Marley and C. E. Floyd Jr., "Application of the Mutual Information Criterion for Feature Selection in Computer-Aided Diagnosis," *Medical Physics*, Vol. 28, No. 12, 2001, pp. 2394-2402.

J. Software Engineering & Applications, 2010, 3, 933-938
doi:10.4236/jsea.2010.310110 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Software Architecture and Methodology as a Tool for Efficient Software Engineering Process: A Critical Appraisal

Achimugu Philip[1], Babajide Afolabi[2], Oluwaranti Adeniran[2], Gambo Ishaya[2], Oluwagbemi Oluwatolani[1]

[1]Computer Science Department, Lead City University, Ibadan; [2]Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife.
Email: {check4philo, igpeni, tolapeace}@yahoo.com, {bafox, aranti}@oauife.edu.ng

Received July 10th, 2010; revised August 14th, 2010; accepted August 18th, 2010.

ABSTRACT

The foundation for any software system is its architecture. Software architecture is a view of the system that includes the system's major components, the behaviour of those components as visible to the rest of the system, and the ways in which the components interact and coordinate to achieve the overall system's goal. Every efficient software system arises as a result of sound architectural basement. This requires the use of good architecture engineering practices and methods. This paper recognizes software architecture practice as a discipline pervading all phases of software development and also presents an enhanced model for software engineering process which provides an avenue for speedy, efficient and timely delivery of software products to their intended users. The integration of software architecture into the phases of software development process in a generic software life cycle is also contained in this research report. This is to enable software engineers and system analysts to use effective software architecture practices and to employ appropriate methodology during the software engineering process.

Keywords: *Software Systems, Architecture, Software Engineering, System Life Cycle, Software Components*

1. Introduction

Many people limit the term software engineering to just computer program. In the real sense of it, it is not just the program but also the associated documentation and design principles required to make these programs operate correctly. Software products may be developed for a particular customer or for general market, so they undergo series of thoughts and ideas that account for their initial inception, development, production, operation, upkeep and usability from one generation to another [1].

Software engineering process or activities therefore can be considered as sets of activities and associated results which produce a software product. They include software specification, development, validation and evolution. Software process model represents a networked sequence of activities, objects, transformations and events that embodies strategies for accomplishing software evolution. Different process models organize these activities in different ways, in different level of details and they are best suited for different project complexities.

Software architecture and methodology practice has emerged as a crucial part of the design process and is the main focus of this paper. Software architecture encompasses the structures of large software systems. The architectural view of a system is abstract, distilling away details of implementation, algorithm, and data representation and concentrating on the behaviour and interaction of "black box" elements. Software architecture is developed as the first step toward designing a system that has a collection of desired properties [2,3] put it very nicely in this formula (Software architecture = {Elements, Forms, Rationale/Constraints}).

Software methodology on the other hand, is a pre-defined sequence of events that must be executed, followed or carried out in order to produce a well structured and robust software product that meets user's requirement and produce good scalable tendencies.

Therefore, this paper argues that software engineers who have sound knowledge of software architecture and appropriate methodology to be employed in software engineering process will be better informed and hence produce good quality software and deliver same at the appropriate time, thus avoiding breach of contract which is common amongst software engineers.

2. Software Architecture Practice

Today, software architecture practice is one sub-discipline within software engineering that is concerned with the high-level (abstract) design of the software of one or more systems. Software architecture are created, evolved, and maintained in a complex environment. The architecture business cycle [1] of **Figure 1** illustrates this. On the left hand side, the figure presents different factors that influence a software architecture through an architect. It is the responsibility of the architect to manage these factors and take care of the architecture of the system. An important factor is formed by requirements, which come from stakeholders and the developing organization. The architect also has the capacity of influencing opinions of stakeholders, refine user's requirement in a way that it captures all the activities of an organization as well as determine the technicalities of the proposed software in terms of development techniques, architectural considerations, programming language (s) to be used and the extent of scalability of the database.

2.1. What is Architectural during Software Engineering Process?

During software development, what is architectural can be determined based on what architecture is use for. The criterion for something to be architectural is this: It must be a component, or a relationship between components, or a property (of components or relationships) that needs to be externally visible in order to reason about the ability of the system to meet its quality requirements or to support decomposition of the system into independently implementable pieces. The following are some corollaries of this principle:

1) *Architecture describes what is in your system.* When you have determined your context, you have determined a boundary that describes what is in and what is out of your system (which might be someone else's subsystem). Architecture describes the part that is in.

2) *Architecture is an abstract depiction of your system.* The information in an architecture is the most abstract and yet meaningful depiction of that aspect of the system. Given the architectural specification, there should not be a need for a more abstract description. That is not to say that all aspects of architecture are abstract, nor is it to say that there is an abstraction threshold that needs to be exceeded before a piece of design information can be considered architectural.

3) *What's architectural should be critical for reasoning about critical requirements.* The architecture bridges the gap between requirements and the rest of the design. If you feel that some information is critical for reasoning about how your system will meet its requirements then it is architectural. You, as the architect, are the best judge. On the other hand, if you can eliminate some details and still compose a forceful argument through models, simulation, walk-throughs, and so on about how your architecture will satisfy key requirements then those details do not belong. However, if you put too much detail into

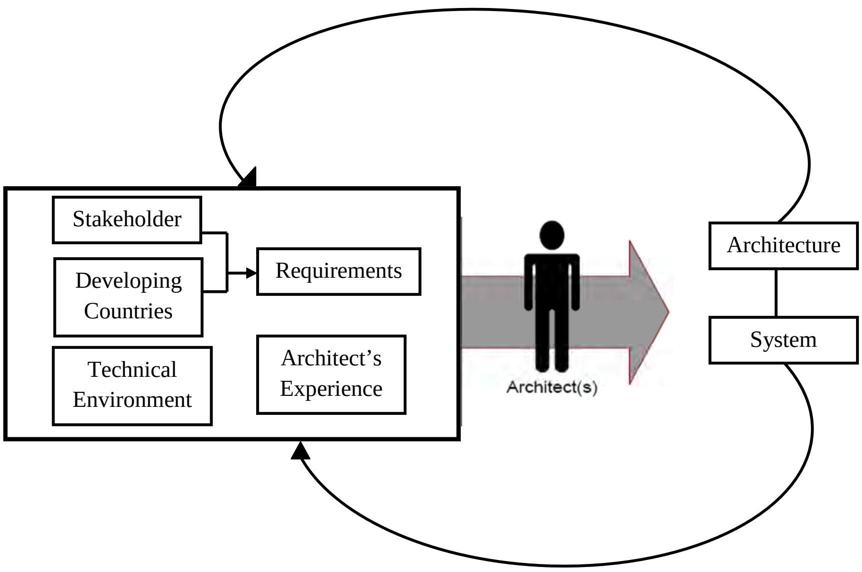


Figure 1. The architecture business cycle (Source: Bass *et al.*, 2003).

your architecture then it might not satisfy the next principle.

4) *An architectural specification needs to be graspable.* The whole point of a gross-level system depiction is that you can understand it and reason about it. Too much detail will defeat this purpose.

5) *Architecture is constraining.* It imposes requirements on all lower-level design specifications. It's good to distinguish between when a decision is made and when it is realized. For example, one can determine a process prioritization strategy, a component redundancy strategy, or a set of encapsulation rules when designing architecture; but might not actually make priority assignments, determine the algorithm for a redundant calculation, or specify the details of an interface until much later.

Generally, what is architectural is the most abstract depiction of the system that enables reasoning about critical requirements and constrains all subsequent refinements.

2.2. Integrating Software Architecture Practice into Software Development Process

Software architecture practice can be integrated into all the phases of software development methodologies and models [4]. This is used to distinguish it from particular analysis and design methodologies. Since the architecture determines the quality of the system, it then makes a lot of sense to have architectural design built into the software development process [5]. As shown in the **Figure 2**, software architecture is integrated into all the phases in development process. The role of software architecture in each phase of the software development process is established. The model shows that during the requirements phase of development, an architecture may be used to identify, prioritize, and record system concerns and desires. During design and analysis, an architecture may be used to model, visualize, and analyze design decisions chosen to address the principal concerns and achieve the desired qualities. Decisions may be guided by adopting one or more architectural styles. During implementation and testing, an architecture may be used to drive testing, instantiate a product, support runtime dynamism, or enforce security policies. Rather than throwing out an architecture at this point as is often done, an architecture remains part of the product. During maintenance, an architecture may be used as a basis for incorporating new features, or increasing modelling detail.

3. Methodology in Software Engineering Process

[6] defines software development methodology as the framework that is used to structure, plan and control the process of developing a software product or information systems. A wide variety of such frameworks has evolved over the years, each with its own recognized strength and

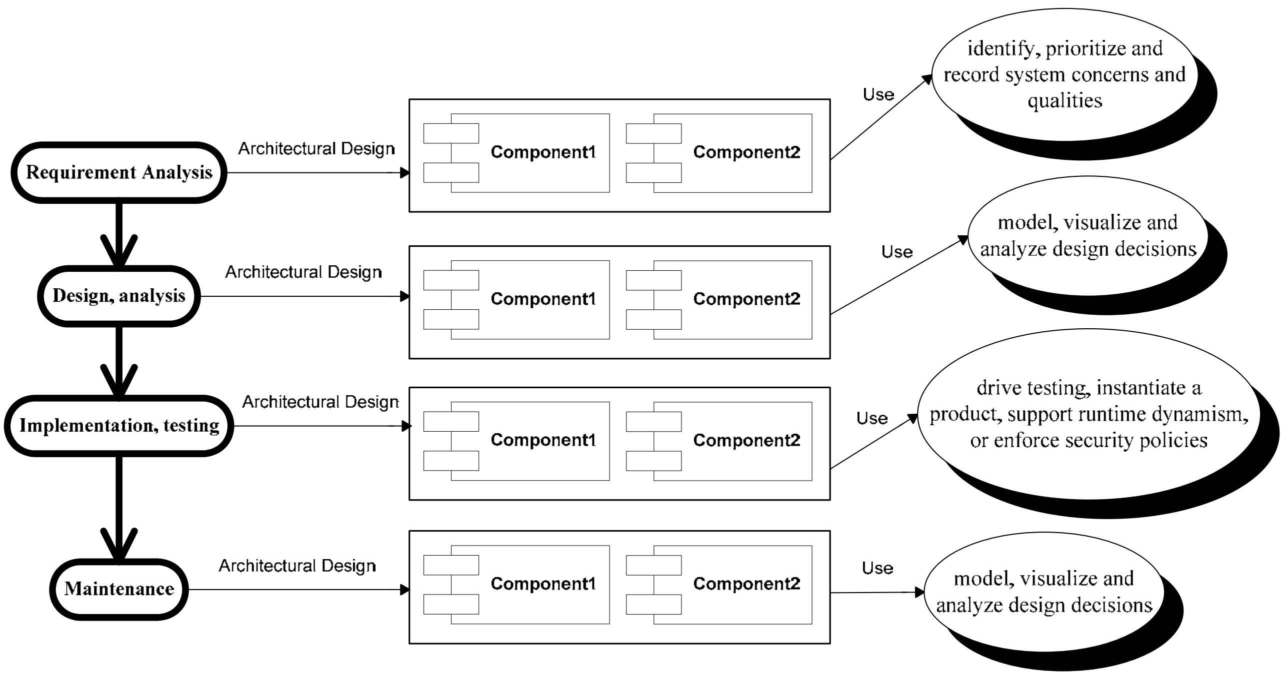


Figure 2. A model integrating architecture into software development process.

weaknesses. One system development methodology is not necessarily suitable for use by all projects. Each of the available methodologies is best suited for specific kinds of projects, based on various technical, organizational, project and team considerations. The framework of a software development methodology consists of:

1) A software development philosophy with the approach or approaches of the software development process.

2) Multiple tools, models and methods, to assist in the software development process.

These frameworks are often bound to some kind of organization, which further develops, support the use, and promotes the methodology. The methodology is often documented in some kind of formal documentation. This section therefore presents recommendations of appropriate methodology to be used in the software engineering process. This work is based on the theoretical study of some existing software process models. These models were ranked based on the following features:

1) Ease of use and management
2) Support for small projects
3) Support for complex projects
4) Adequate test plan
5) Support for dynamic user requirement
6) Risk analysis
7) Early delivery of project
8) Level of requirements gathered
9) Cost effectiveness
10) Meeting user's need
11) Activity based
12) Deliverable based

[7] asserts that ease of use and management implies that each phase of the development process has a specific deliverable and the documentation of this makes it easy to manage. Support for project complexities (small or complex) implies effectiveness of the model when used for different projects. How and when testing is done is of great significance in the development process of software product. It implies whether it is done at the beginning, end of development or at end of each phase. In most cases, user's requirements are dynamic, so how well a model adjust to this dynamism is important. Also [8] argued that the level of user's requirements gathered that is, detailed or scanty, at the beginning phase, plays a great deal in whether the product will adequately meet user's needs. A model that adapts well with changing user requirements tends to meet users needs better. Cost effectiveness is a relative term when used in software engineering because there is always a trade-off between cash and kind implications, so this was not used to rank the models considered but its importance was not thrown away. It worth mentioning that these rankings are based entirely on findings from books and articles as referenced. The model with the best rank for each feature considered was adopted to form this optimal model.

The model that ranks highest is finally adopted for each feature in our model. Activities that lead to the achievement of these desired features are identified and the proposed model will emphasize them. This serves as the bases of the adoption. However, **Table 1** shows the various types of models and when they are best at use.

Table 1. Re-ranking of the models with best rank.

features	waterfall	incremental	Rapid prototyping	v-shape	spiral	JAD	Object process
Ease of use and management	Best			good			Better
Support for small project	Better			Best			
Support for complex project		better	good		Better	better	Best
Test plan			better	Best		good	Better
Support for dynamic requirement		Better	Better			good	Best
Risk analysis					Best	Better	
Early delivery of project		Better	Best				Good
Level of requirement gathered	Better			good	Better	Better	Best
Cost effective							
Meeting user need		Better	Best			good	Better
Activity based	Yes			Yes	Yes	Yes	
Delivery based		Yes	Yes				Yes

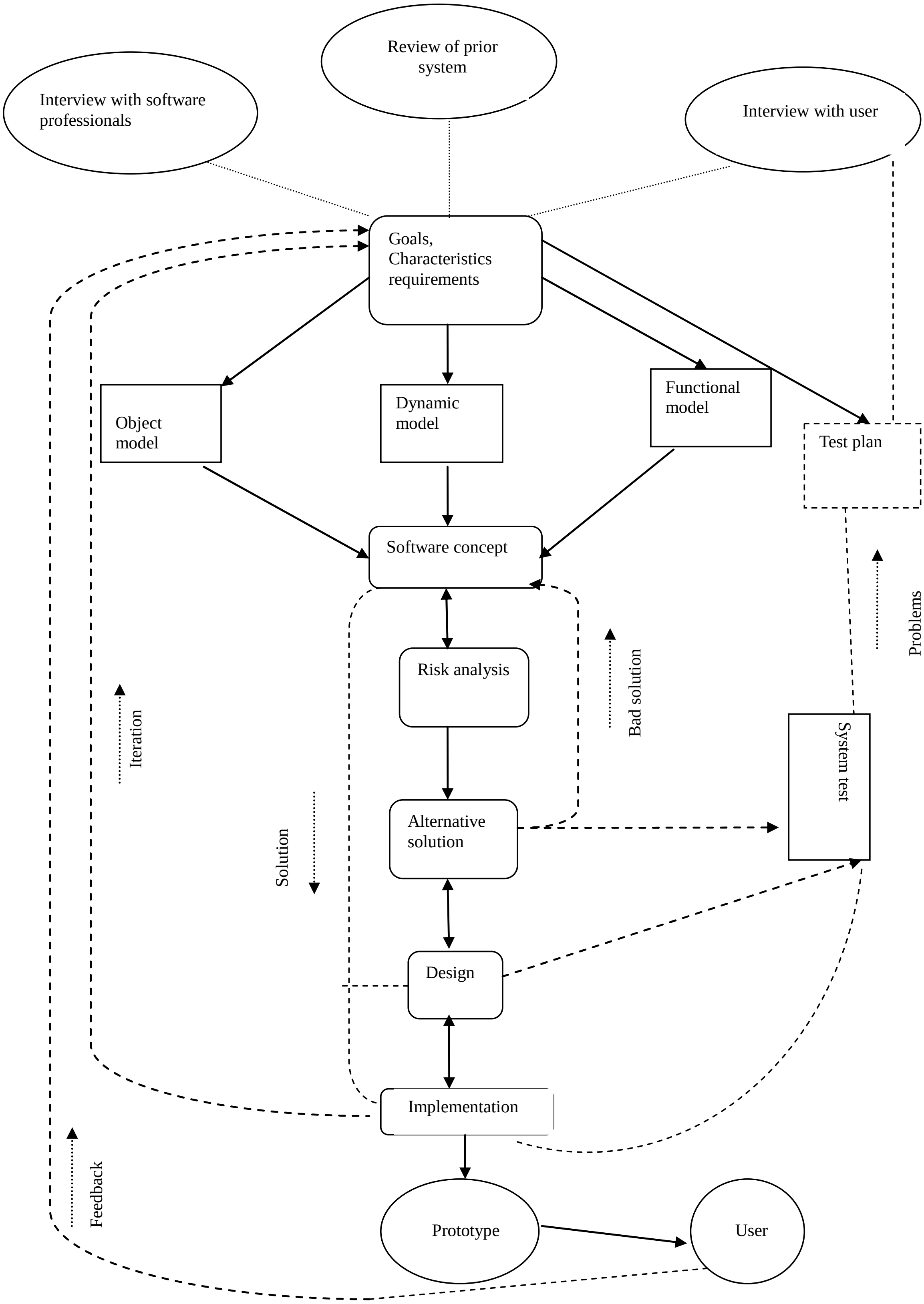


Figure 3. Enhanced model.

Figure 3 gives a pictorial description of the model. Basically the model is easy to use and manage because at the end of each phase, there is a specific deliverable and a review of the process involved. Also the phases cascade like the waterfall model but to ensure that it is not as rigid as waterfall model, the phases go back and forward that is, if there is problem in one stage, it documented and kept for next iteration where the stage will be revisited. Testing is done early before coding is done and at the end of each phase, a test plan is created to ensure quality delivery. It uses concept from object oriented analysis, design and programming to ensure support for different project complexities. [9] reviewed that deliverable at the end of analysis phase is considered objects with attributes and methods; they also have constructors which are methods describing how to create the deliverable and quality assurance methods. Naturally user's requirements are dynamic, the object oriented approach of this model allows for this to be defined in a single deliverable called "task context" which can be modified without affecting the entire production process. After gathering the user requirement, a process is undertaken to identify the risk and alternate solutions. A prototype is produced at the end of this phase and this ensures that the product is delivered early to the user though with reduced functionalities. Feedback from users is used to provide a better and user oriented software. Cost effectiveness can be viewed as optimal cost for optimal solution, so this model can be said to be cost effective.

Clearly seen from the figure, requirements gathered are expanded into three views; object view represents the artefacts of the system, dynamic view represents the interaction between objects, and functional view represents methods of the system. This is the object oriented approach of the model. The phases cascade and iterate so; problems found during testing are adequately taken care of in the next iteration which corresponds to an improved version of prototype. No throwaway prototype is developed in this model because of the risk analysis which gives rise to alternate solutions.

4. Conclusions

System developers and acquirers can use effective software architecture practices across the life cycle to ensure predictable product qualities, cost, and schedule. We establish in this paper that software architecture is the bridge between mission/business goals and a software system. Secondly, software architecture drives software development throughout the life cycle, and finally the paper identifies some methodologies that could be employed during the software engineering process using some parameters. An enhanced model for software engineering process was also proposed.

REFERENCES

[1] Bass *et al.*, "Software Architecture in Practice," Addison Wesley, New York, 2003.

[2] P. Clements, "Predicting Software Quality by Architecture-Level Evaluation," *Proceedings of the Fifth International Conference on Software Quality*, Austin, Texas, October 1995.

[3] D. L. Parnas, "Designing Software for Ease of Extension and Contraction," *IEEE Transactions on Software Engineering*, Vol. 5, No. 2, 1979, pp. 128-138.

[4] B. A. Boehm, "Spiral Models of Software development and Enhancement," *Computer*, Vol. 20, No. 9, 1987, pp. 61-72.

[5] D. Garlan, "Software Architecture: A Road Map," *Proceeding of the* 16*th International Conference on Software Engineering*, Sorrento, Italy, May 2000, pp. 71-80.

[6] M. M. Lehman, "Process Models, Process Programming, programming Support," *Proceedings of the* 9*th International Conference on Software Engineering IEEE Computer Society*, Amsterdam, 1987, pp. 14-16.

[7] R. Lewallen, "Software Development Life Cycle Models," *Net Development*, Vol. 20, No. 9, 2006, pp. 61-72.

[8] W. W. Royce, "Managing the Development of Large Software Systems," *proceedings of the* 9*th International Conference Software Engineering, IEEE Computer Society*, Amsterdam, 1987, pp. 328-338.

[9] W. Scacchi, "Process Models in Software Engineering," *Encyclopaedia of Software Engineering*, 2nd Edition, John Wiley and Sons, Inc., New York, 2001.

J. Software Engineering & Applications, 2010, 3, 939-943
doi:10.4236/jsea.2010.310111 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Digital Image Watermarking Algorithm Based on Fast Curvelet Transform

Jindong Xu, Huimin Pang, Jianping Zhao

College of Physical Engineering, Qufu Normal University, Shandong, China.
Email: xujindong1980@yahoo.com.cn

Received July 15th, 2010; revised August 18th, 2010; accepted August 21st, 2010.

ABSTRACT

A digital image watermarking algorithm based on fast curvelet transform is proposed. Firstly, the carrier image is decomposed by fast curvelet transform, and, the watermarking image is scrambled by Arnold transform. Secondly, the binary watermarking image is embedded into the medium frequency coefficients according to the human visual characteristics and curvelet coefficients. Experiment results show that the proposed algorithm has good performance in both invisibility and security and also has good robustness against the noise, cropping, filtering, JPEG compression and other attacks.

Keywords: *Digital Image Watermarking, Fast Curvelet Transform, Human Visual Characters, Robustness, Invisibility*

1. Introduction

In the past two decades, more and more researchers have devoted to the transform domain which shows good performance and robustness. Recently, Candès and Donoho [1,2] have developed a new multiscale transform which is called the Curvelet transform. It is anisotropic with strong direction, and provides optimally sparse representations of objects along a general curve with bounded curvature. Hence, it has been widely used in image processing field. At the same time, many research papers proposed to embed watermark based on the first curvelet transform [3-8].

Shi *et al.* [3] proposed a semi-fragile watermarking algorithm by embedding the wartermark in the maximum module of curvelet coefficient. This algorithm keeps good tolerance against JPEG compression. The method of Thai *et al.* [4] embedded a watermark in curvelet coefficients which are selected by a threshold. This method has good invisibility and robustness. Thai *et al.* [5] embedded the watermark in the curvelet transform which contain as much edge information as possible. It has good invisibility but poor robustness. The implementation of the first curvelet transform includes subband decomposition, smooth partitioning, renormalize and ridgelet analysis. So these algorithms have many problems such as implementation complexity and large amount of redundancy.

This paper proposes a digital image watermarking algorithm based on fast curvelet transform. Firstly, we take a binary-valued image which contains copyright information as watermark, and adapt Arnold transform to it to improve its security. Secondly, according to the human visual characteristics, we embed the watermark in curvelet coefficients of the original image. This algorithm is simple and easy to implement. Experiment results show that our algorithm has good invisibility and security, and is robustness to the noise, cropping, filtering, JPEG compression and other attacks.

2. Curvelet Transform

Similar to the wavelet transform and the ridgelet transform, the curvelet transform theory is based on sparsity theory [8]. The idea of curvelet is to calculate the inner relationship between the signal and the curvelet function to realize the sparse representation of the signal.

2.1. Continuous—Time Curvelet Transform

The curvelet transform can be expressed as

$$c(j,l,k) := \left\langle f, \varphi_{j,l,k} \right\rangle \tag{1}$$

here, $j = 0, 1, 2, \ldots$, is a scale parameter; $l = 0, 1, 2, \ldots$, is an orientation parameter; and $k = (k_1, k_2) \in \mathrm{Z}^2$ is a translation parameter. The mother curvelet is $\varphi_j(x)$, its Fourier transform is $\varphi_j(\omega) = U_j(\omega)$, where U_j is frequency window defined in the polar coordinate system such as:

$$U_j(r,\theta) = 2^{-3j/4} W\left(2^{-j} r\right) V\left(\frac{2\lfloor j/2 \rfloor \theta}{2\pi}\right) \tag{2}$$

W and V are radial and angular windows respectively and will always obey certain admissibility conditions. Curvelet at scale 2^{-j}, orientation θ_l and position $x_k^{(j,l)} = R_{\theta_l}^{-1}\left(k_1 \times 2^{-j}, k_2 \times 2^{-j}\right)$ can be expressed as:

$$\phi_{j,l,k}(x) = \phi_j\left[R_{\theta_l}\left(x - x_k^{j,l}\right)\right] \tag{3}$$

So, for $f \in L^2(R^2)$, curvelet transform is expressed as:

$$\begin{aligned} c(i,l,k) &:= \frac{1}{2\pi^2} \int \hat{f}(\omega) \overline{\hat{\phi}}_{j,l,k}(\omega) \mathrm{d}\omega \\ &= \frac{1}{2\pi^2} \int \hat{f}(\omega) U_j\left(R_{\theta_l}\omega\right) \exp\left(\mathrm{i}\left\langle x_k^{j,l}, \omega\right\rangle\right) \mathrm{d}\omega \end{aligned} \tag{4}$$

2.2. Digital Curvelet Transform

Digital curvelet transform is linear and takes as input Cartesian arrays of the form $f[t_1, t_2]$, $0 \le t_1, t_2 < n$, which allows the output as a collection of coefficients:

$$c^D(j,l,k) := \sum_{0 \le t_1, t_2 < n} f[t_1, t_2] \overline{\varphi_{j,l,k}^D[t_1, t_2]} \tag{5}$$

In order to improve the curvelet transform—in the sense that they are conceptually simpler, faster and far less redundant. Paper [2] proposed the Fast Discrete Curvelet Transform (FDCT). There are two digital implementation of FDCT. The first is based on unequally-spaced fast Fourier transform (USFFT) while the second is based on the wrapping of specially selected Fourier samples. The FDCT-Wrapping uses simpler choice of spatial grid to translate curvelets at each scale and angle. It needs less two-dimensional FFTs than FDCT-USFFT, so it is quickly.

The architecture of FDCT via Wrapping is then roughly as follows:

1) Apply the 2D FFT and obtain Fourier sample $\hat{f}[n_1, n_2]$, $-n/2 \le n_1, n_2 \le n/2$ (n is the size of the picture).

2) For each scale/angle pair (j, l), form the product $\tilde{U}_{j,l}[n_1, n_2] \hat{f}[n_1, n_2]$.

3) Wrap this product around the origin and obtain $\tilde{f}_{j,l}[n_1, n_2] = W\left(\tilde{U}_{j,l} \hat{f}\right)[n_1, n_2]$, where the range for n_1 and n_2 is now $0 \le n_1 < L_{1,j}$ and $0 \le n_2 < L_{2,j}$ (for θ in the range $(-\pi/4, \pi/4)$).

4) Apply the inverse 2D FFT to each $\tilde{f}_{j,l}$, hence collecting the discrete coefficients $c^D(j,l,k)$.

3. Watermarking Algorithm Based on FDCT

3.1. Watermark

In this algorithm, the watermark is a binary-valued image with 32 × 32 pixels (**Figure 1**).

Arnold transform is used to increase the data security, and its function is defined as (6).

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ k & k+1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \bmod N \tag{6}$$

Adapt n (here, $n = 8$) times Arnold transform to the original watermark W, then we obtain scrambling watermark (**Figure 2**).

3.2. Watermark Embedding

According to the Human Visual Characteristic, we choose to embed the watermark into medium-frequency. In selected scale, the coefficients of each orientation are sorted and find criterion T_i by amplitude factor λ ($\lambda \ge 0$). Then choose the minimum T_i as embedding parameter T. Finally, the watermark is embedded to coefficients which are chosen by T .

The embedding procedure (**Figure 3**) is described as follows.

1) Apply FDCT to the original image, get curvelet coefficient C.

2) Select the curvelet coefficient to embed a bit watermark according to the following conditions.

$$\begin{aligned} &\text{For 0} \quad 0 < C(j,l,k) < T/2 \quad \text{or} \quad T < C(j,l,k) < 3T/2 \\ &\text{For 1} \quad T/2 < C(j,l,k) < T \end{aligned} \tag{7}$$

Then record the positions of these coefficients and the positions of 0 and 1 in the watermark.

3) The embedded coefficients are modified by the following equations.

$$\begin{aligned} \text{Embed 0} \quad C'(j,l,k) &= C(j,l,k) \\ &\quad - \operatorname{mod}\left(C(j,l,k), T\right) + T/4 \\ \text{Embed 1} \quad C'(j,l,k) &= C(j,l,k) \\ &\quad - \operatorname{mod}\left(C(j,l,k), T\right) + 3T/4 \end{aligned} \tag{8}$$

4) Do inverse FDCT to C', obtain the watermarked image.

3.3. Watermark Extraction

The extraction procedure is composed of 4 steps and each step is described as follows.

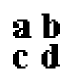


Figure 1. Watermark.

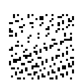


Figure 2. Scrambled watermark.

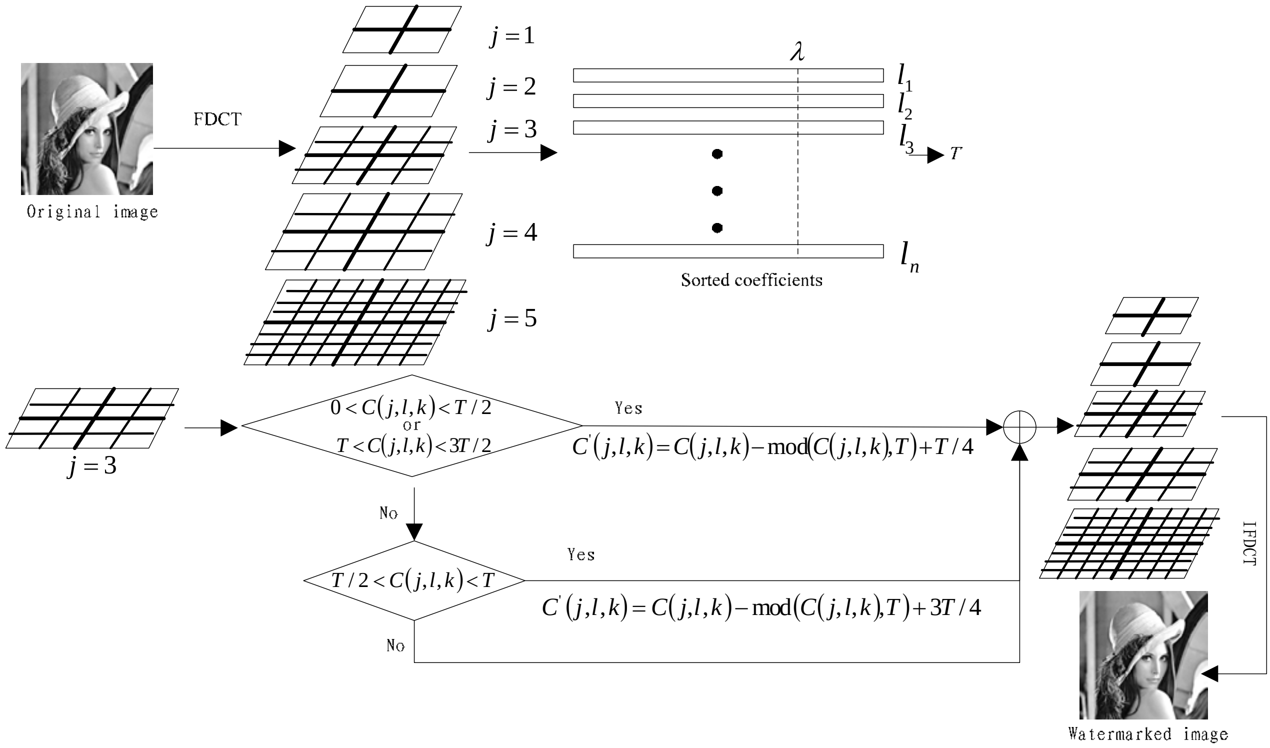


Figure 3. Embedding procedure.

1) Apply curvelet transform to watermarked image, obtain curvelet coefficient *C*'.

2) Locate the watermark positions from the original image by using embedding procedure step 2. Then extract the coefficients *C*' from the watermarked image using those watermark position.

3) Extract the watermark W' from C' with the following rule

$$\omega_i = \begin{cases} 1 & if \quad \mathrm{mod}\left(C'\left(l, j, k\right), T\right) \geq T/2 \\ 0 & if \quad \mathrm{mod}\left(C'\left(l, j, k\right), T\right) < T/2 \end{cases} \qquad (9)$$

4) Apply *T*'-*n* (*T*' is Arnold transform period) times Arnold transform to W', obtain the binary watermark image.

4. Experimental Results

In this program, the image is transformed through FDCT via Wrapping. We use standard 512 × 512 pixel image 'Lena' (**Figure 4**) for evaluation of our proposed method, and conduct experiments binary-valued image watermarking with the noted above parameter scale = 3, λ = 0.75.

We have investigated the invisibility and the robustness of our watermarking system, analyzed the algorithm performance by objective and subjective standards.

4.1. Invisibility Tests

The watermarked image is shown in **Figure 5(a)**, the detected watermark from watermarked image is shown in **Figure 5(b)**.

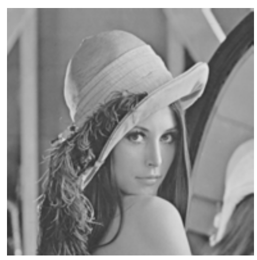
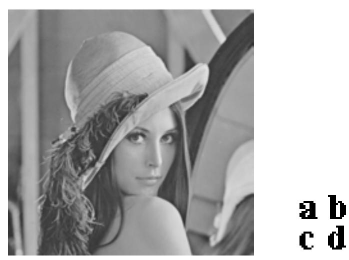


Figure 4. Original image.



(a) (b)

Figure 5. Invisibility Tests. (a) Watermarked image [PSNR = 60.8028dB]; (b) Detected watermark [NC = 1.00].

Table 1. JPEG compression attack.

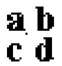
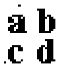
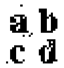
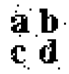
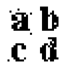
Q	90	80	70	60	50	40	30	20	**15**
PSNR	25.2093	25.1357	25.0710	25.0186	24.9718	24.8890	24.8226	24.6714	24.4813
NC	0.9988	0.9975	0.9901	0.9877	0.9914	0.9741	0.9532	0.9470	0.9113
Detected watermark									

Table 2. Gaussian low pass filtering attack.

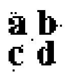
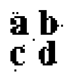
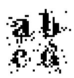
Standard Deviation (*Window*)	0.5(3)	1.5(3)	0.5(5)	1.5(5)	**3(5)**
PSNR	40.8289	32.4277	40.8027	29.9117	28.7562
NC	0.9914	0.9704	0.9914	0.9113	0.8658
Detected watermark					

Table 3. Other attacks.

Attacks	*Gaussian noise* (0.001)	*Salt & Pepper noise* (0.01)	*Random cropping* (1/16)	*Random cropping* (1/32)	**Change contrast**
PSNR	19.4926	25.0252	19.6377	21.7762	25.2690
NC	0.8116	0.9200	0.8645	0.9372	0.8756
Detected watermark					

Subjectively, the watermarked image has good invisibility. Compare **Figure 5(b)** with **Figure 2**, they are fully consistent. The original watermark is accurately recovered.

4.2. Robustness Tests

To evaluate the robustness of algorithm, all the attacks are tested by the software of Stirmark [10,11]. The Peak Signal to Noise Ratio (PSNR) is employed to evaluate the quality of watermarked image after attack, and the Normalized Correlation Coefficient (NC) is used to evaluate the quality of extracted watermark for some attacks such as JPEG compression (Quality factor Q), Gaussian low pass filtering, adding noise, cropping and so on. The simulation results are shown as following Tables.

It can be seen from **Table 1** and **Table 2**, the proposed method show very good robustness to JPEG compression and Gaussian low pass filtering. From **Table 3**, the method also has good robustness against noise, cropping and so on.

5. Conclusions

This paper proposes a method by embedding a watermark into the original image based on FDCT. At the same time, Arnold transform is applied to improve the security of the system. The experimental results show that the watermarked image has good invisibility and robustness against JPEG compression and Gaussian low pass filtering.

There are many unstable coefficients are discovered from the experiments. For instance, a small change of the image will arouse big changes of these coefficients. These unstable factors can influence the extracting of watermark. Therefore, the proposed algorithm can not give a good performance of rotation and scaling, etc. In the future, we will unceasing devote ourselves to the study of the robustness watermark system against geometric attacks based on curvelet transform.

REFERENCES

[1] D. L. Donoho and M. R. Duncan, "Digital Curvelet

Transform: Strategy, Implementation and Experiments," *Proceeding of the SPIE on Wavelet Application VII*, Orlando, 2000, pp. 12-30.

[2] E. J. Candes, L. Demanet and D. L. Donoho, "Fast Discrete Curvelet Transforms [R]," *Applied and Computational Mathematics*, 2005, pp. 1-43.

[3] J. P. Shi and Z. J. Zhai, "Curvelet Transform for Image Authentication," *Rough Set and Knowledge Technology, LNAI* 4062, 2006, pp. 659-664.

[4] T. D. Hien, I. Kei, H. Harak, *et al*., "Curvelet-Domain Image Watermarking Based on Edge-Embedding," *Lecture Notes in Computer Science*, Vol. 4693, No. 2010, 2007, pp. 311-317.

[5] T. D. Hien, K. Miyara, *et al.*, "Curvelet Transform Based Logo Watermarking," *Innovative Algo-rithms and Techniques in Automation, Industrial Electronics and Telecommunication*, 2007, pp. 305-309.

[6] H. Y. Leung, L. M. Cheng and L. L. Cheng, "Digital Watermarking Schemes Using Multi-resolution Curvelet and HVS Model," *Lecture Notes in Computer Science*, Vol. 5703, 2009, pp. 4-13.

[7] H. Y. Leung, L. M. Cheng and L. L. Cheng. "A Robust Watermarking Scheme Using Selective Curvelet Coefficients," *International Conference on Intelligient Information Hiding and Multimedia Signal Processing*, Harbin, August 2008, pp. 465-468.

[8] T. D. Hien, K. Miyara, I. Kei and F. F. Ali, "Digital Watermarking Based on Curvelet Transform," 9*th International Symposium on Signal Processing and Its Applications*, Sharjah, Feburary 2007, pp. 1-4.

[9] J. L. Cheng and T. Shan. "Development and Prospect of Image Multiscale Geometric Analysis," *Acta Electronic Sinica*, Vol. 31, No. 12 A, pp. 1975-1981.

[10] A. P. Fabien and Petitcolas. "Watermarking Schemes Evaluation," *IEEE. Signal Processing*, Vol. 17, No. 5, 2000, pp. 58-64.

[11] Stirmark, http://www.petitcolas.net/fabien/watermarking/stirmark/

J. Software Engineering & Applications, **2010, 3, 944-964**
doi:10.4236/jsea.2010.310112 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Application of Artificial Neural Network, Kriging, and Inverse Distance Weighting Models for Estimation of Scour Depth around Bridge Pier with Bed Sill

Homayoon Seyed Rahman, Keshavarzi Alireza*, Gazni Reza

Water Department, Shiraz University, Shiraz, Iran.
Email: keshavrz@shirazu.ac.ir

Received July 23rd, 2010; revised August 23rd, 2010; accepted August 25th, 2010.

ABSTRACT

This paper outlines the application of the multi-layer perceptron artificial neural network (ANN), ordinary kriging (OK), and inverse distance weighting (IDW) models in the estimation of local scour depth around bridge piers. As part of this study, bridge piers were installed with bed sills at the bed of an experimental flume. Experimental tests were conducted under different flow conditions and varying distances between bridge pier and bed sill. The ANN, OK and IDW models were applied to the experimental data and it was shown that the artificial neural network model predicts local scour depth more accurately than the kriging and inverse distance weighting models. It was found that the ANN with two hidden layers was the optimum model to predict local scour depth. The results from the sixth test case showed that the ANN with one hidden layer and 17 hidden nodes was the best model to predict local scour depth. Whereas the results from the fifth test case found that the ANN with three hidden layers was the best model to predict local scour depth.

Keywords*: *Artificial Neural Network, Scour Depth, Ordinary Kriging, Inverse Distance Weighting, Bridge Piers, Bed Sill

1. Introduction

The accurate estimation of maximum scour depth around and downstream of bridge piers is critical and very important for design engineers. The prediction of scour depth around bridge piers has been the subject of many experimental studies, and has resulted in a number of prediction techniques being presented. Scour depth is a significant limiting factor when assigning the minimum depth of substructures, as it decreases the lateral capacity of the substructure.

To determine a technique for predicting scour depth for different pier positions, comprehensive experimental tests have been conducted. In the past, a number of research studies had been conducted to determine techniques for the estimation of local scour around bridge piers and their abutments. These have been reported in literature. Of these studies, the first extensive experimental work on bridge pier scour was conducted and reported by Chabert and Engeldinger (1956) which is cited by Jeng *et al.* [1]. A study was also conducted by Blodgett (1978), again is cited by Jeng *et al.* [1], to report the cause of failure of 383 bridges. It was reported that most failures were caused by catastrophic floods. This study also found that the incorrect prediction of local scour depth during engineering design lead to enlarged local and contraction scour. Yankielun and Zabilansky [2] pointed out that this serious problem costs millions of dollars worth of damage, leaving foundations of bridge piers and bridge abutments insecure. Johnson [3] compared some of proposed prediction methods with the available field data, and concluded that more research is still required to accurately determine local scour.

To overcome this complicated problem, the artificial neural network (ANN) was found to be useful as a comprehensive function approximator, especially when the relationship between dependent and independent variables is inadequately understood [1]. Trent *et al.* [4,5]

*Present Address: School of Civil & Environmental Engineering, UTS, Sydney, Australia.

applied ANN to estimate local pier scour and sediment transport in open channels. Choi and Cheong [6] estimated local scour around bridge piers using ANN and concluded that the ANN can successfully predict the depth of scour over a wider range of conditions with a greater accuracy than existing empirical formulae.

Butcher [7] pointed out that the kriging methods of geostatistical analysis provide valuable techniques for the analysis of sediment contamination problems, including interpolation of concentration maps from point data and the estimation of global mean concentrations. Biglari and Sturm [8] pointed out that bridge failure due to local scour around piers and abutments has motivated many examinations into scour prediction, as well as reliable design methods. Liriano and Day [9] compared current prediction equations for culvert outlets with results obtained from two ANN models. They concluded that the ANN model can be used to predict local scour in laboratory and in the field better than other empirical relationships that are currently in use.

Kambekar and Deo [10] analyzed scour data using different neural network models that were developed to predict scour depth. They found that the neural network provides a better alternative to statistical curve fitting. Jeng *et al.* [1], Bateni *et al.* [11] and Lee *et al.* [12] applied neural networks to predict scour depths around bridge piers. Bateni *et al.* [13] used Bayesian neural networks for the prediction of equilibrium and time-dependent scour depth around bridge piers. They showed that the new models estimate equilibrium and time-dependent scour depth more accurately than the existing expressions.

The results of training and testing ANN obtained from these models have been analyzed and an accurate model to predict local scour depth around a bridge pier in a river environment has been produced. (see **Figures 1** and **2**) These results contribute to the understanding of local scour and provide engineers with a way of determining scour depth for a variety of pier situations.

In most previous studies, scouring was studied around bridge piers installed without the presence of a bed sill. This paper presents the experimental data used to investigate bridge scour around a pier installed upstream of a bed sill and explores the use of artificial neural networks, kriging and inverse distance weighting models to estimate the scour depth around a bridge pier.

2. Experiment Setup and Procedure

A dimensional analysis was conducted to find the most important parameters for bed scouring around a curved bed sill installed downstream of a bridge pier. The most effective parameters were found to be;

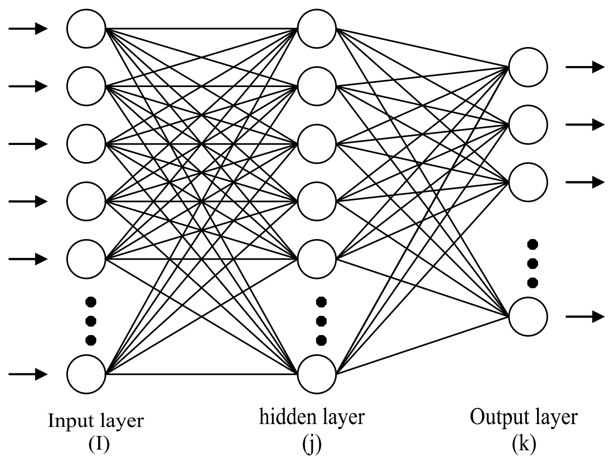


Figure 1. Conceptual diagram of a feed forward network with one hidden layer.

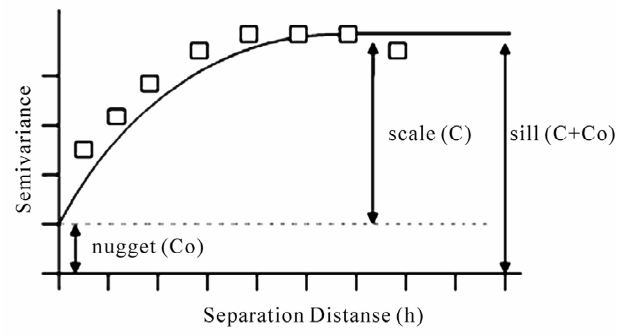


Figure 2. Variogram parameters.

$$Y_S = f(g, \nu, \rho_w, \rho_s, q, W, d_{50}, r, D) \quad (1)$$

in which Y_s is the scouring depth, g is the acceleration of gravity, ρ_w is the flow density, ρ_s is the particle density, q is the flow discharge per unit width, d_{50} is the median particle diameter, W is the width of the channel, r is the arch distance of the circular sill and D is sill diameter. In this experimental study the *r/W* and *D/W* are investigated only during the laboratory experiments.

The laboratory experiments were carried out with 50 mm diameter circular piers installed at different distances from bed sill. The sill height was 12 cm and it was installed in a 15 m long, 0.5 m wide, 0.5 m deep experimental flume in the Hydraulic Laboratory College of Agriculture, Shiraz University. The scour depth and flow depth were measured using a sandy surface meter. The experiment setup and pier installation are shown in **Figure 3**.

The length and width of the scour were measured after each experiment. The longitudinal profile and maximum

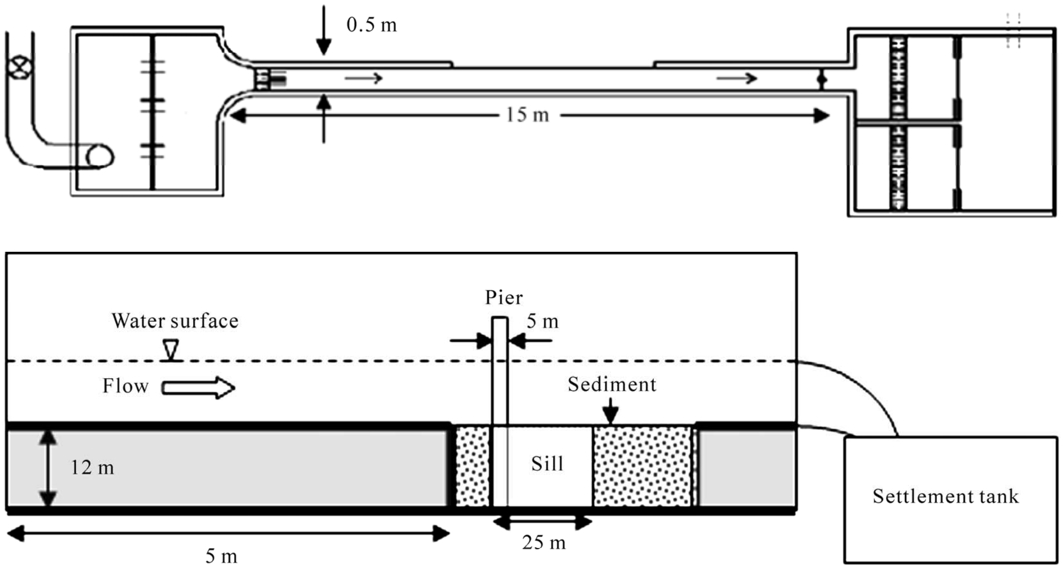


Figure 3. Experimental setup and pier installation.

depth of the scour were measured during the experiments. The scour measured on the side and at the rear of the pier were approximately the same, whilst the scour measured at the nose of the pier was less than the back scour.

A false bottom was installed in the flume to create a recess for the sediment bed. The recess was 2.5 m long, 0.5 m wide and 0.12 m deep and filled with non-cohesive sediment with D_{50} equal to 0.5 mm diameter and standard deviation equal to 1.23 mm. The pier was installed firstly by preparing an undersized pilot hole, pushing the pier in, and then trimming off the remaining sediment. Water entered the flume smoothly from an inlet reservoir, and a sediment trap was used at the downstream end of the test reach. Downstream of the trap, water passed over a tailgate into a sump. Water depth in the in-floor flume was controlled by a tail gate located at the downstream end of the test section. All the experiments were conducted under steady flow conditions. The flow discharge was measured by a 90 degree V-Notch and an electromagnetic flow meter.

Experimental test cases were conducted in eight different bed sill models (**Figure 4**). **Table 1** explains the flow condition of the experimental tests.

3. Results and Discussions

3.1. The ANN, OK and IDW Estimations for Scour around Piers

The whole data set, consisting of 2754 data points, was composed of eight different conditions. Each condition was divided into two parts randomly: a training set consisting of 80% of the data points and a validation or testing set consisting of 20% of the data points.

In this study, two types of ANN models were developed: 1) single hidden-layer ANN model consisting of only one hidden layer, and 2) multiple hidden-layer ANN model consisting of two and three hidden layers. The task of identifying the number of neurons in the input and output layers is usually simple, as it is dictated by the input and output variables considered to model the physical process.

As previously mentioned, the number of neurons in the hidden layer(s) can be determined through the use of trial and error procedure [1]. The optimal architecture was determined by varying the number of hidden neurons (from 1 to 20), and then the best structure was selected. The training of the ANN models was stopped when either the acceptable level of error was achieved or when the number of iterations exceeded a prescribed maximum of 2500. The learning rate of 0.05 was also used.

ANN was implemented using the MATLAB software package (MATLAB version 7.2 with neural network toolboxes) [14].

The performance of ANN, OK and IDW configurations were assessed based on calculating the mean absolute error (MAE), and the root mean square error (RMSE). (see **Table 2**).

The coefficient of determination, R^2 of linear regression line, between the predicted values from each method and the desired output were also used as a measure of performance. The three statistical parameters used to compare the performance of the various method con-

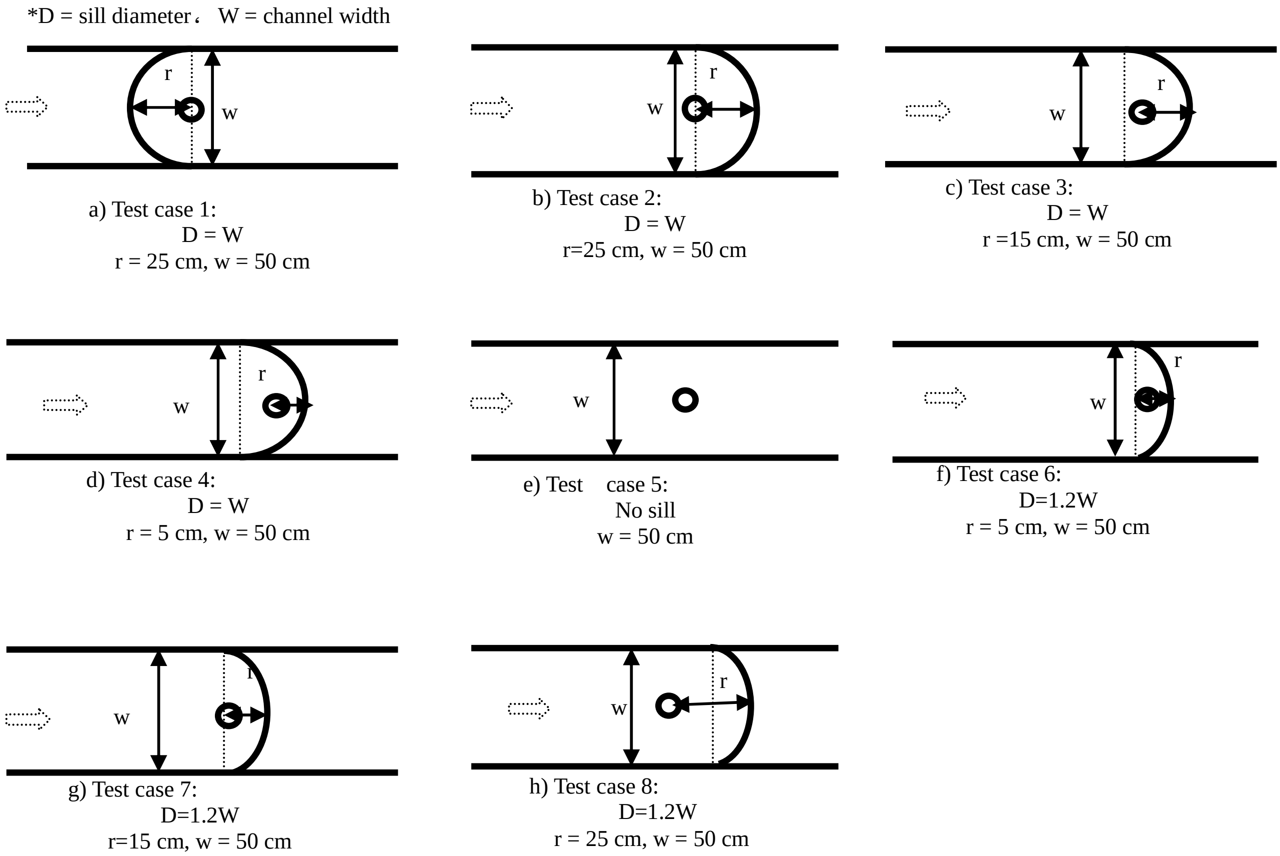


Figure 4. Schematic configuration of eight bed sill models used in this study.

Table 1. Hydraulic condition and geometric parameters of the experimental tests.

Test Case	D/W*	Discharge (lit/sec)	Head water (cm)	sill radius (cm)	Velocity (m/s)	F_r	Pier diameter (cm)
1	1	13.3	9	25	0.2955	0.31	5
2	1	12.64	7.9	25	0.32	0.363	5
3	1	10	8.3	25	0.241	0.267	5
4	1	11	9.2	25	0.239	0.252	5
5	----	9.5	7.8	----	0.244	0.279	5
6	1.2	9.5	7.6	30	0.25	0.289	5
7	1.2	8.4	7.2	30	0.233	0.277	5
8	1.2	7	5.2	30	0.269	0.377	5

figurations are:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|O_i - t_i\right|, \quad (3)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\left(O_i - t_i\right)^2}{N}}, \quad (4)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(O_i - t_i\right)^2}{\sum_{i=1}^{N}\left(O_i - \bar{O}_i\right)^2} \quad (5)$$

Where: O_i and t_i are observed and predicted for the i th output, and $\bar{O}_i$ is the average of predicted, and N is the total number of events considered.

Table 2. Performance of all test cases.

Test Cases	Methods	Validation			Training		
		MAE	RMSE	R^2	MAE	RMSE	R^2
First test case	ANN	1.93	5.48	0.989	0.46	2.14	0.998
	OK	2.99	6.83	0.954	2.73	5.19	0.957
	IDW	4.31	8.19	0.912	3.74	6.08	0.913
Second test case	ANN	4.92	4.39	0.931	1.71	2.85	0.993
	OK	4.45	4.47	0.927	5.47	5.09	0.861
	IDW	6.90	5.57	0.918	8.37	6.30	0.831
Third test case	ANN	2.62	3.40	0.970	1.64	2.52	0.992
	OK	2.99	3.64	0.945	5.42	4.57	0.844
	IDW	3.91	4.16	0.937	6.18	4.89	0.839
Fourth test case	ANN	4.36	5.10	0.951	3.41	4.21	0.971
	OK	3.95	4.91	0.916	3.27	4.12	0.944
	IDW	4.64	5.32	0.909	3.99	4.56	0.941
Fifth test case	ANN	0.78	4.38	0.997	0.71	3.65	0.998
	OK	1.04	5.05	0.995	1.26	4.88	0.994
	IDW	1.28	5.61	0.991	1.43	5.19	0.992
Sixth test case	ANN	2.37	3.31	0.931	2.58	1.92	0.980
	OK	4.32	4.47	0.930	11.79	4.11	0.609
	IDW	4.99	4.80	0.918	11.80	4.11	0.609
Seventh test case	ANN	2.35	6.78	0.937	1.59	5.16	0.980
	OK	3.65	8.46	0.897	3.78	7.96	0.934
	IDW	3.92	8.76	0.878	2.99	7.08	0.945
Eighth test case	ANN	1.75	3.53	0.987	0.90	2.63	0.998
	OK	2.99	4.62	0.963	3.22	4.98	0.965
	IDW	3.07	4.68	0.961	2.60	4.47	0.969

Cross-validation analysis was used to evaluate effective parameters for OK and IDW interpolations and to compare the different estimation techniques to determine the best approach for accurate prediction data. In cross-validation, each measured point in a spatial domain is individually removed from the domain and its value is estimated by kriging and compared to the actual value as though it were never there (Gamma Design Software [15]).

3.2. First Test Case

In first test case, the tip of the sill was set in the flow direction (in the shape of a convex) and a space of 25 cm was set between the sill and bridge pier. The diameter of the sill is equal to the flume width.

In ANN prediction, optimal architecture is determined by varying the number of hidden neurons (from 1 to 20), and the best structure is selected. It was found that the most accurate results involved use of the feed forward back propagation with two hidden layers and an architecture of configuration: 2-7-4-1.

To evaluate the performance of the ANN, OK and IDW, observed local scour depth values are plotted against the predicted values. **Figure 5** illustrates the results with the performance indices between predicted and observed data for the training and testing data sets, respectively.

Figures 5(c-f) also exhibit that kriging has a lower training error compared with IDW, and its validation error becomes lower than IDW. In other words, kriging validation results have a lower scatter than IDW. As it can be seen from **Figure 5a-b**, ANN has performed well in predicting the local scour depth.

Comparing ANN results with those of OK and IDW it is found that ANN has the lowest training error and validation error, then kriging and IDW. Also interpolated local scour maps (**Figure 6**) show that the interpolated map of the ANN model is more similar to the interpolated map of the observation map as compared with the interpolated map of kriging and IDW.

3.3. Second Test Case

In the second test case, the bed sill was set in the flow direction (in the shape of a concave) and spaces of 25cm were set between the bridge pier and bed sill. **Figure 7** illustrates the results with the performance indices between predicted and observed data for the training and testing data sets, respectively.

When the methods were compared, the training accuracy was significant. It is observed that all models perform with poor accuracy in comparison with results of the first data set. Comparison between these three validations of evaluating scour depth in all runs for ANN, OK and IDW revealed that the difference in accuracy between those was not significant. Also the accuracy of training in OK and IDW was less than the validation. It is shown that these methods are not reliable in this condition because it cannot predict the training data well. In this condition, the ANN model performed well and the IDW had the lowest accuracy. The interpolated maps are demonstrated in **Figure 8**.

3.4. Third Test Case

In a similar manner, the third condition data was used to predict local scour depth with ANN, OK and IDW. In this third test case, similar to second test, the bed sill was set in the flow direction but the distance between the bridge pier and bed sill was set to 15 cm.

Figure 9 shows the results with the performance indices between predicted and observed data for the training and testing data sets, respectively. Again, the ANN model performs well in training and validation. The accuracy of prediction was not considerably different between OK and IDW and as it is shown in **Figure 10**. The interpolated map of observation data, ANN, OK and IDW for the third test case is shown in **Figure 10**.

Accuracy of training was more than validation in the ANN prediction, whereas in the two other mentioned methods, correctness of training was less than the validation. In other words, the results of the OK and IDW were not as precise as for ANN. The accuracy in training of ANN was more reliable when compared with OK and IDW methods.

3.5. Fourth Test Case

The fourth test case was similar to second test, but the distance between bridge pier and bed sill was set to 5cm. From ANN prediction, the best structure was found to be a configuration of 2-4-4-1.

Figure 11 shows the results with the performance indices between predicted and observed data for the training and testing data sets, respectively.

The three interpolated maps of the aforementioned methods are very similar (**Figure 1**2) and comparison between these three training and validations for evaluating scour depth in all runs for ANN, OK and IDW revealed that the difference in accuracy between them was not significant.

3.6. Fifth Test Case

In the fifth test case the bridge pier was set separately in the flume. From ANN estimates, unlike the previous four conditions, the best result was obtained for a 2-4-4-4-1 structure.

The comparison between these three training and validations for evaluating scour depth in all runs for ANN, OK and IDW revealed that the difference in accuracy between these was not significant (**Figure 13**). It was found that the ANN accuracy was better than other methods.

This test was similar to the fourth test where the three interpolated maps of the methods were very similar to interpolated map of observed data (**Figure 14**).

3.7. Sixth Test Case

This experiment was similar to the second test case,

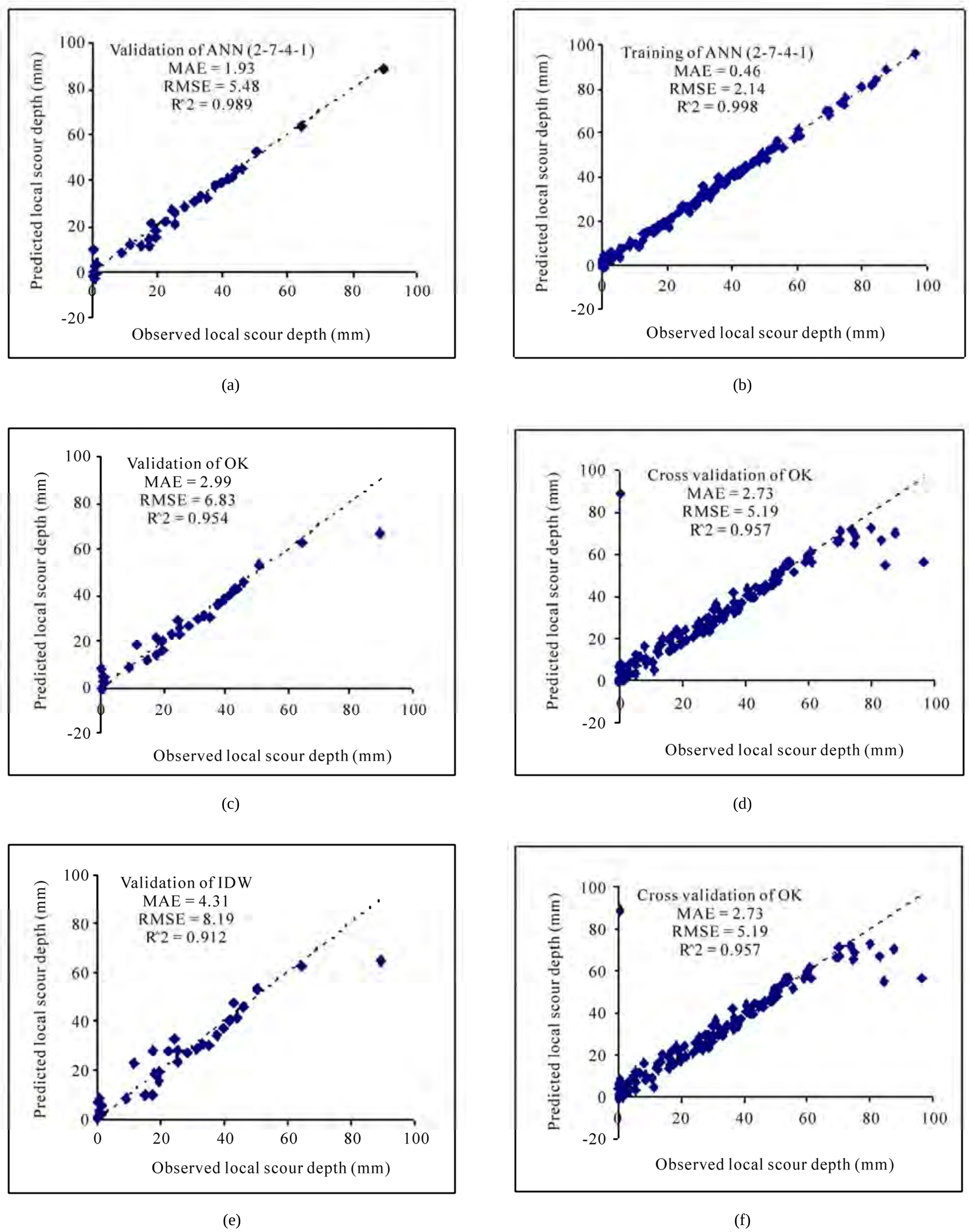


Figure 5. Performance of ANN, OK and IDW for the first test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.

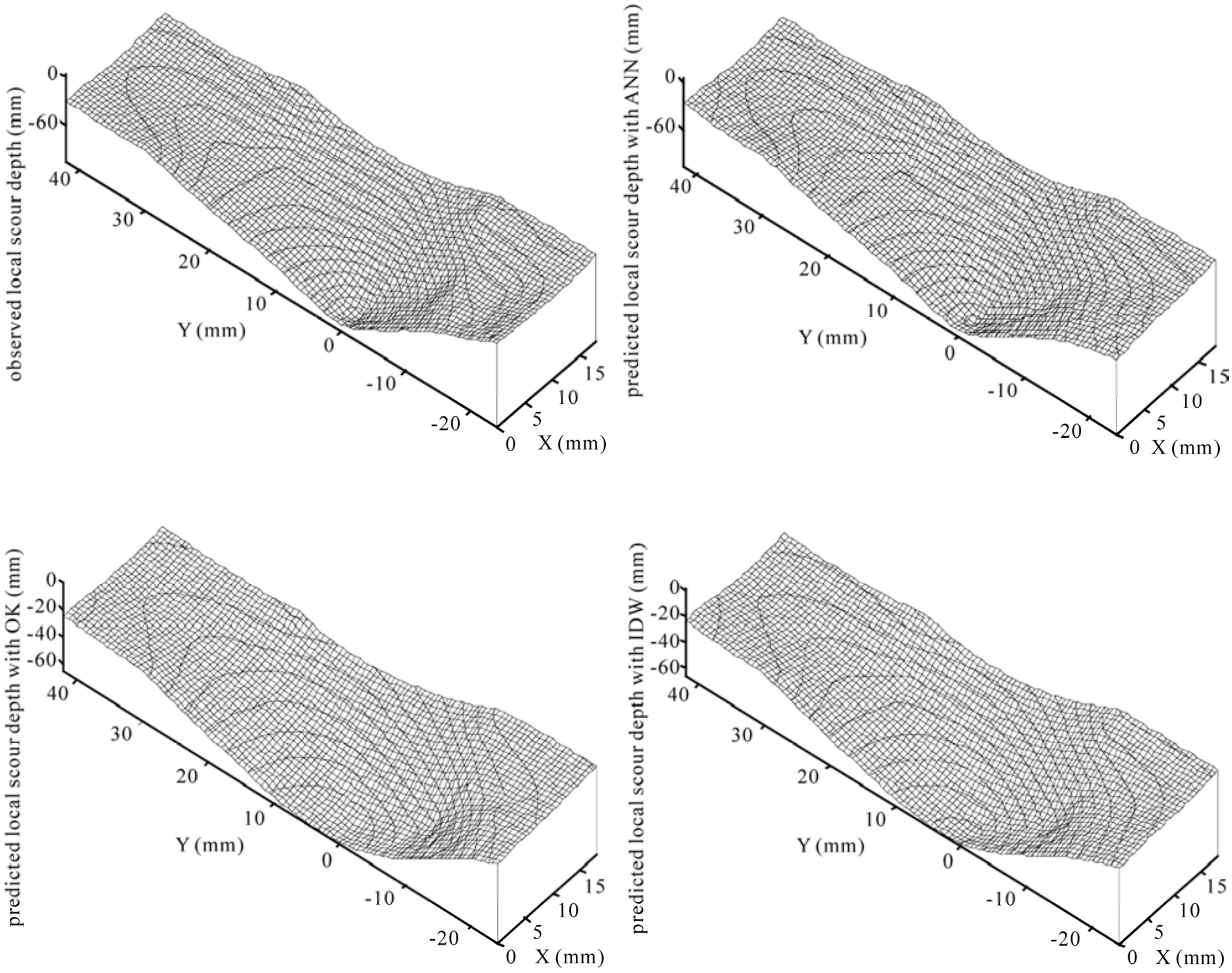
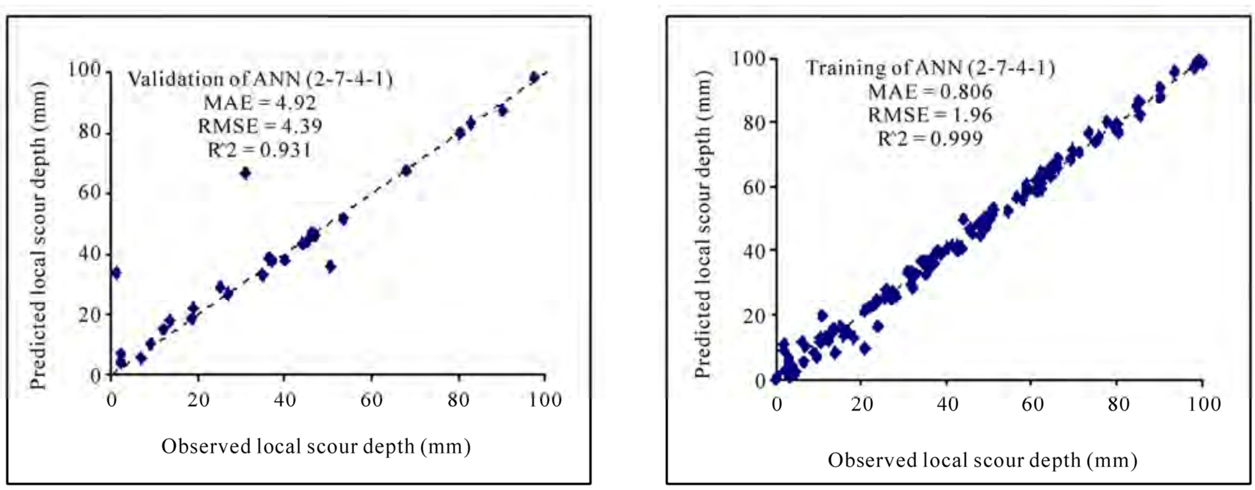
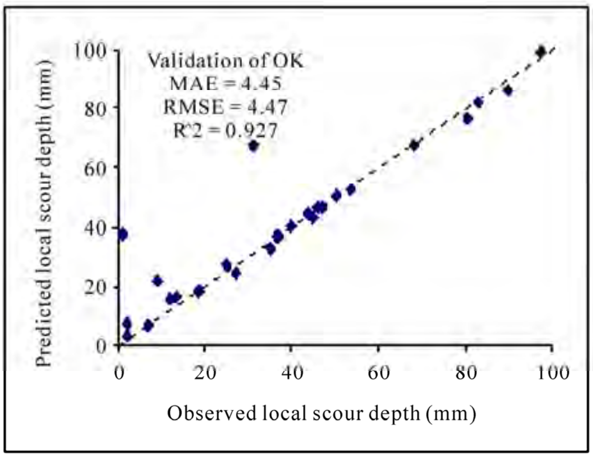


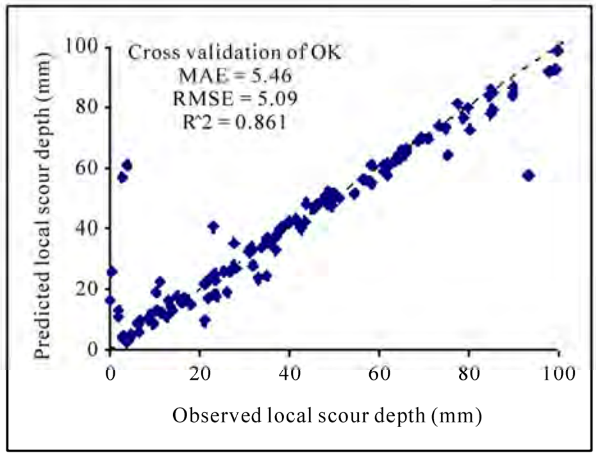
Figure 6. Interpolated maps of observed data, ANN, OK and IDW test for first Test case.



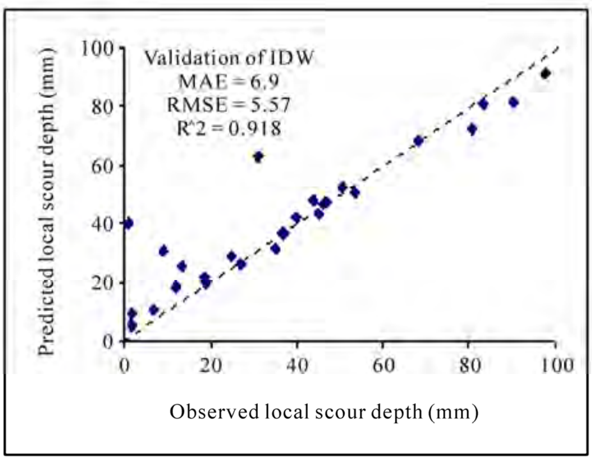
(a) (b)



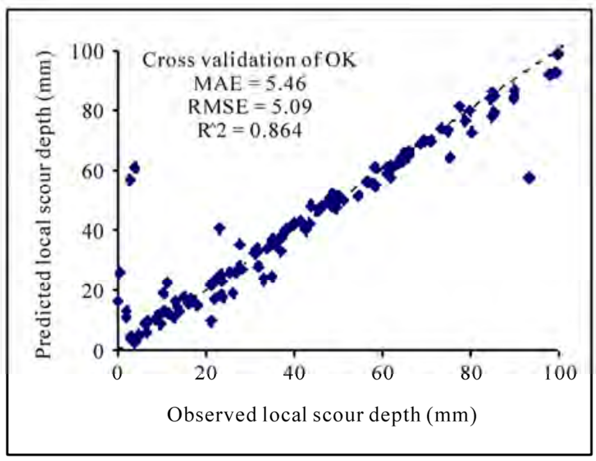
(c)



(d)



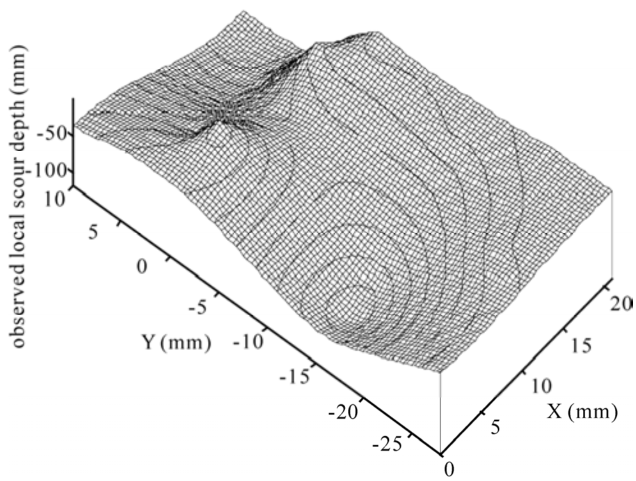
(e)

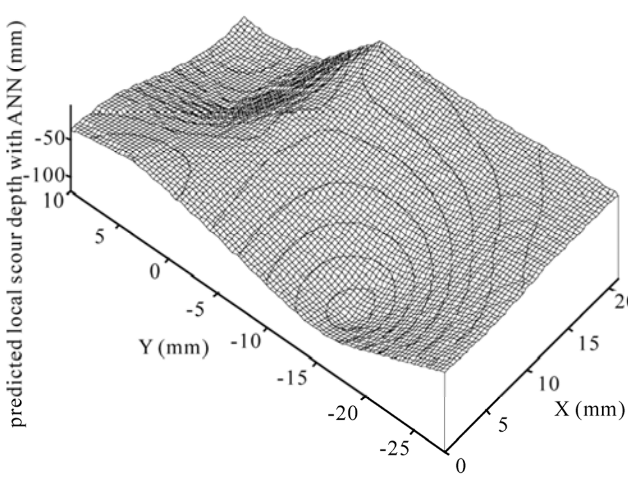


(f)

Performance of ANN of first test case (convex type); a) testing, b) training

Figure 7. Performance of ANN, OK and IDW for the 2nd test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.





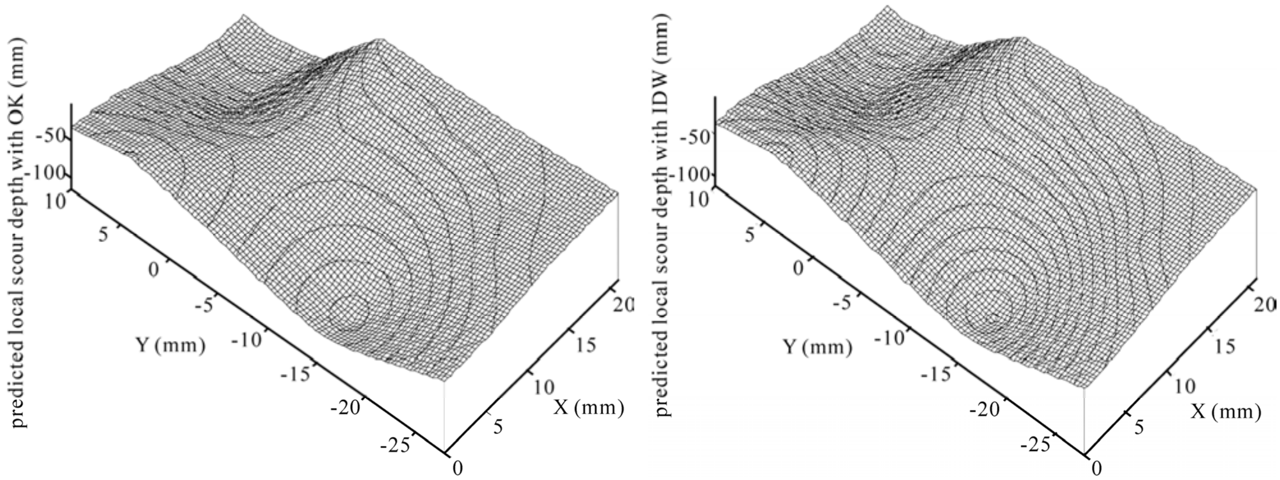
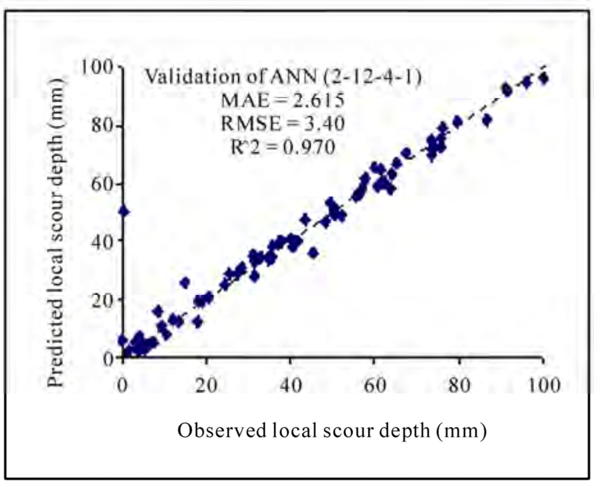
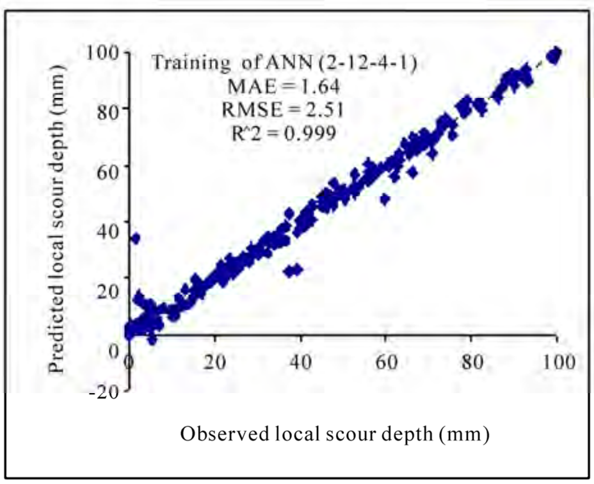


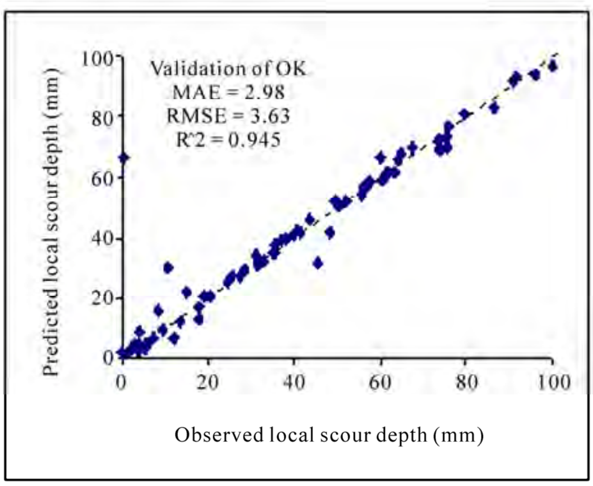
Figure 8. Interpolated maps of observed data, ANN, OK and IDW test for second Test case.



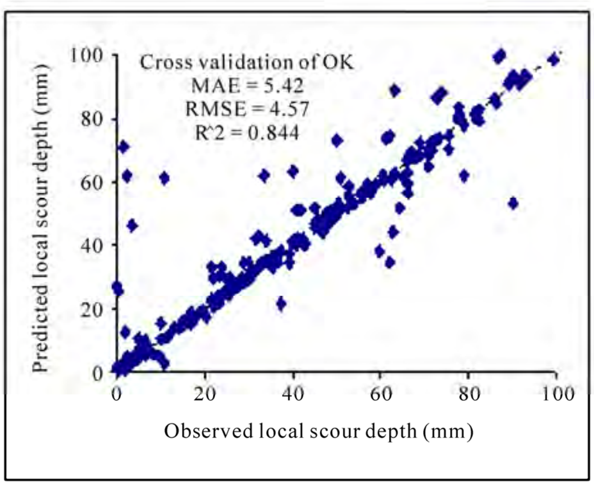
(a)



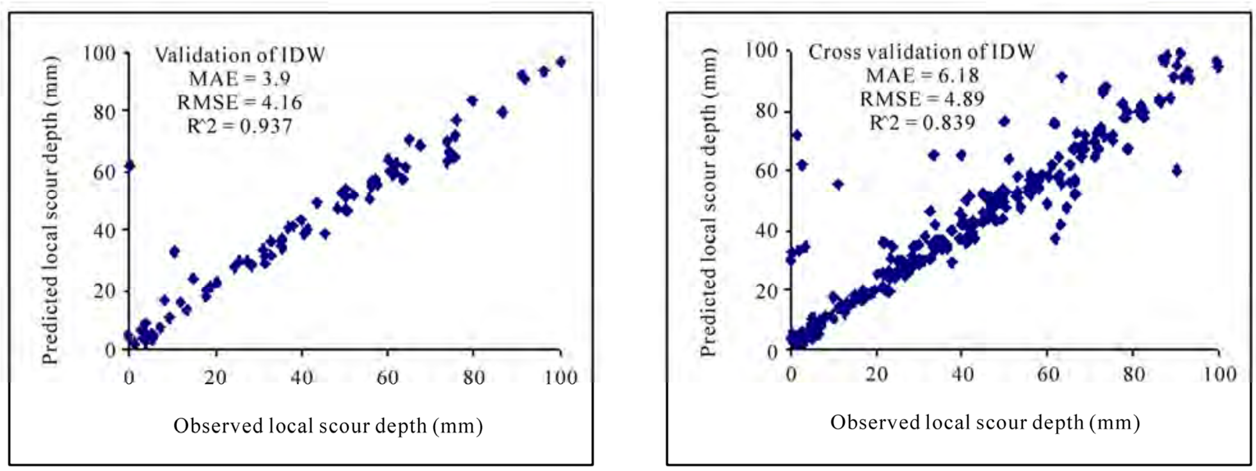
(b)



(c)



(d)



(e) (f)

Figure 9. Performance of ANN, OK and IDW for the third test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.

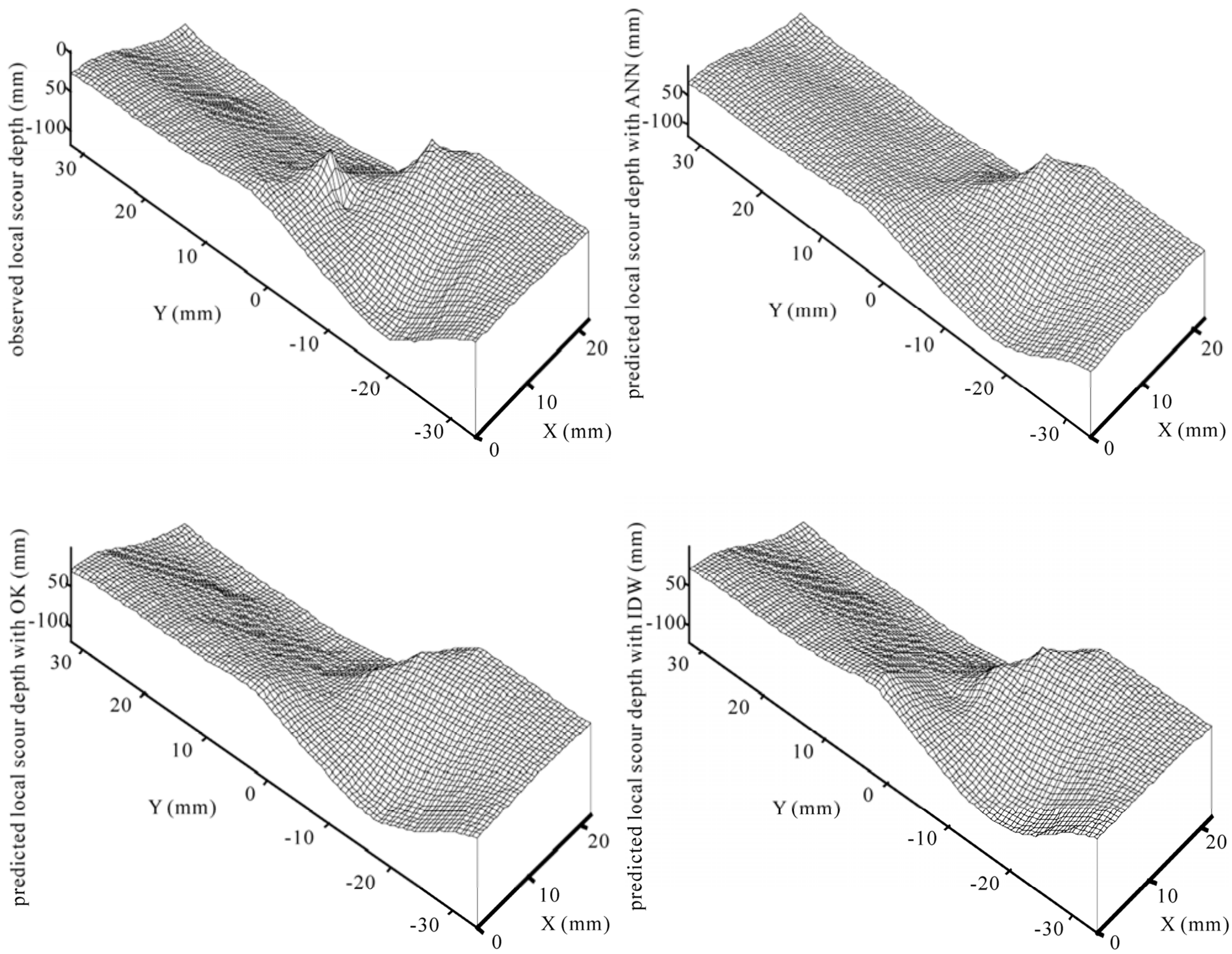


Figure 10. Interpolated maps of observed data, ANN, OK and IDW test for third Test case.

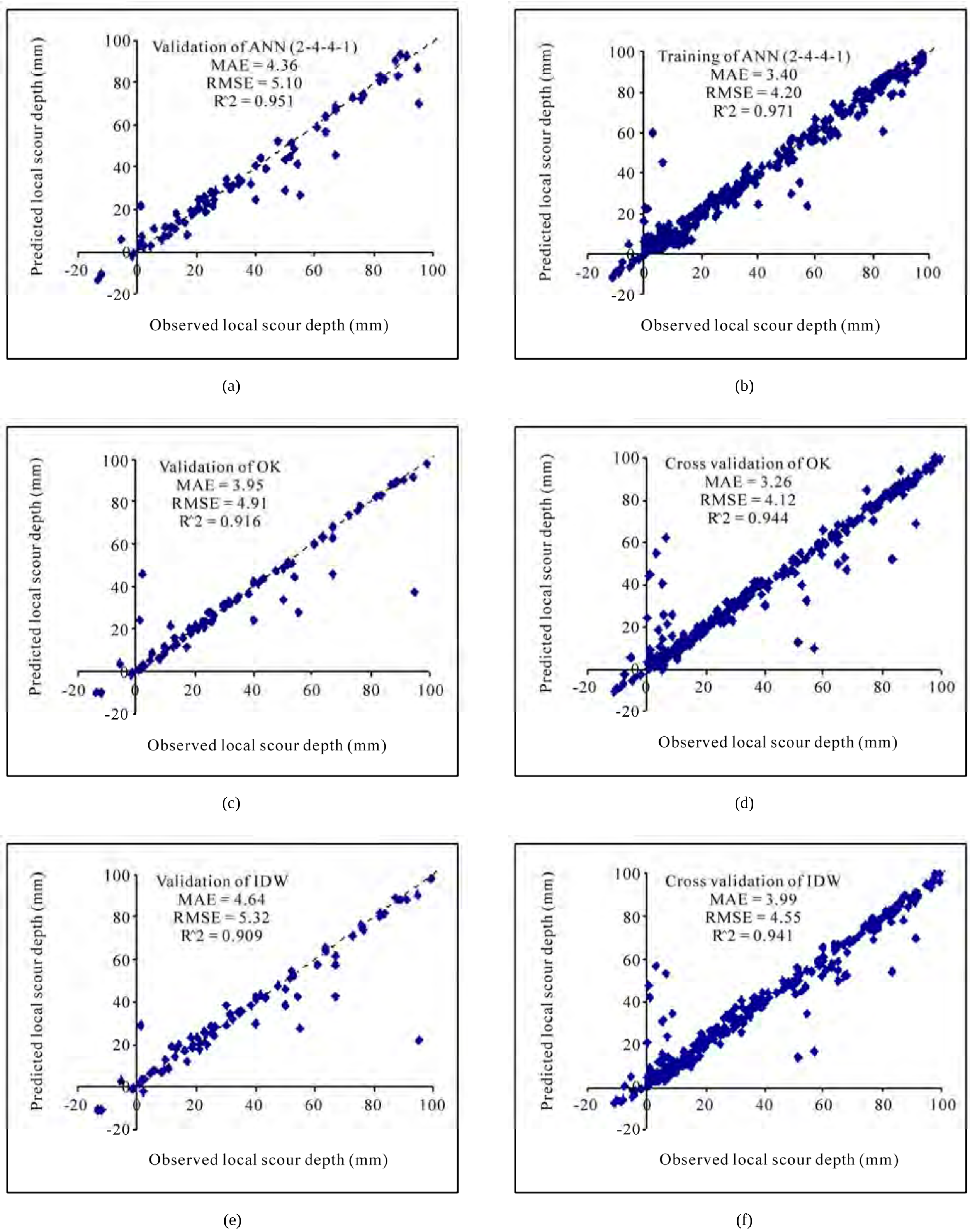


Figure 11. Performance of ANN, OK and IDW for the fourth test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.

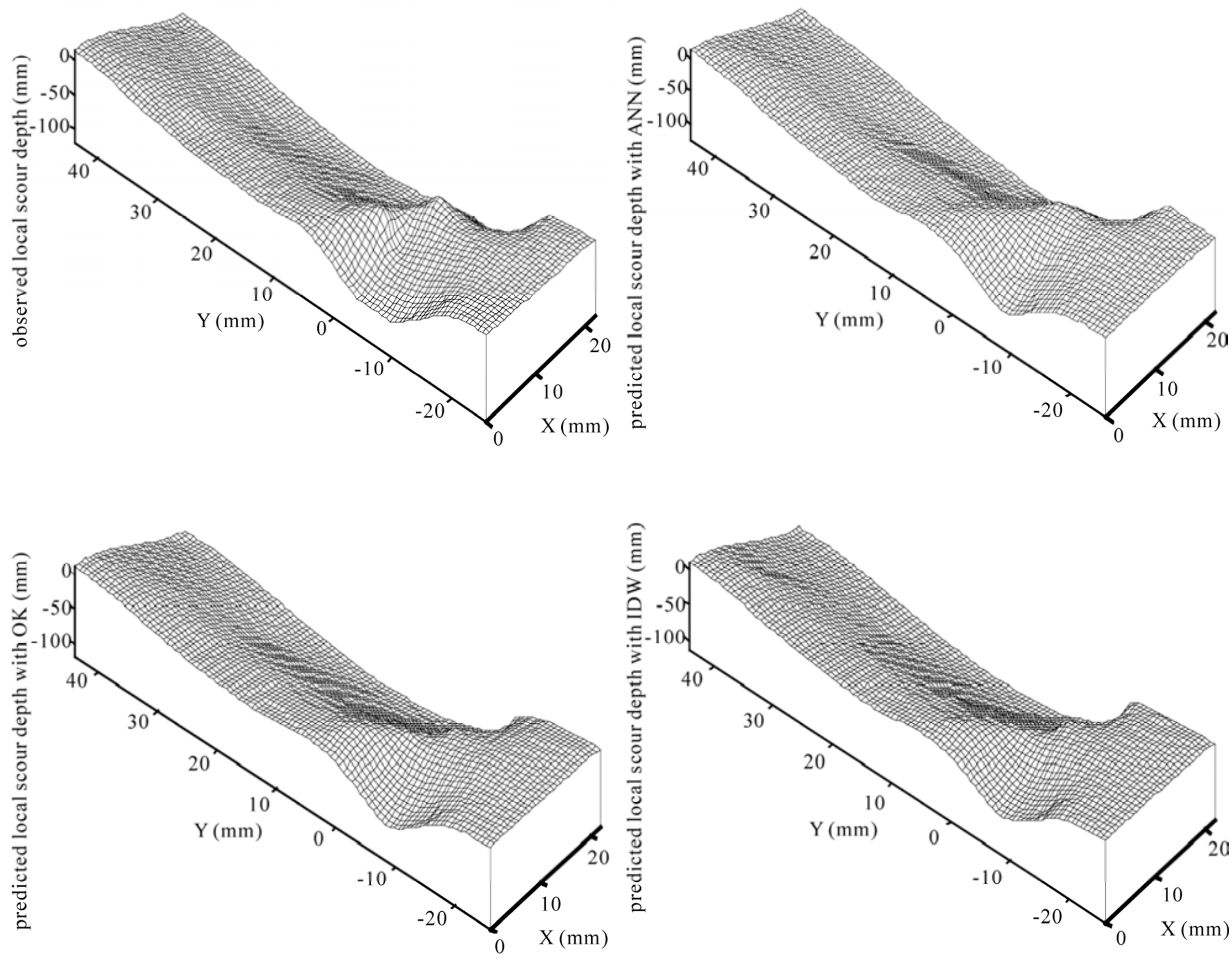
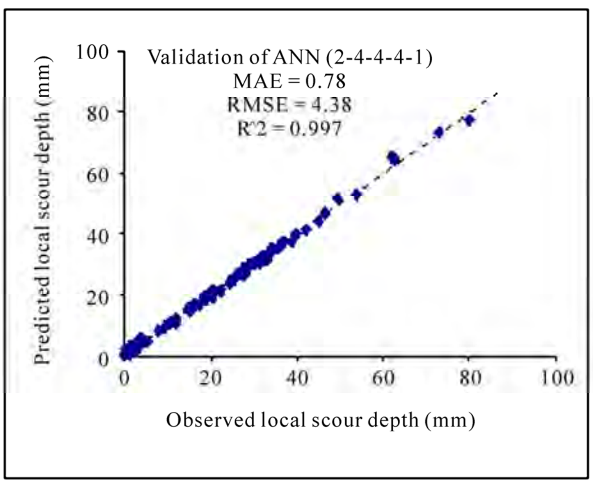
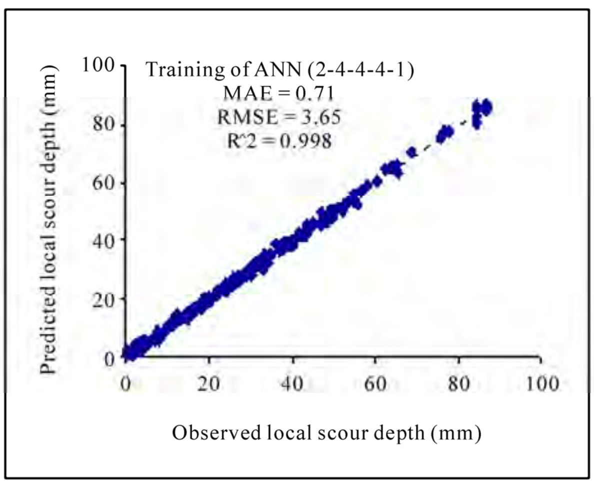


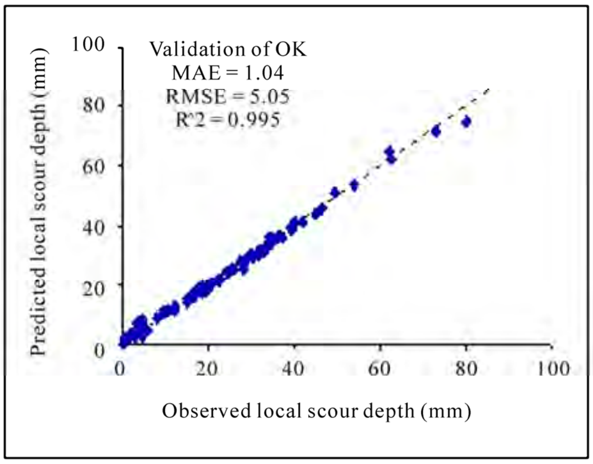
Figure 12. Interpolated maps of observed data, ANN, OK and IDW test for fourth Test case.



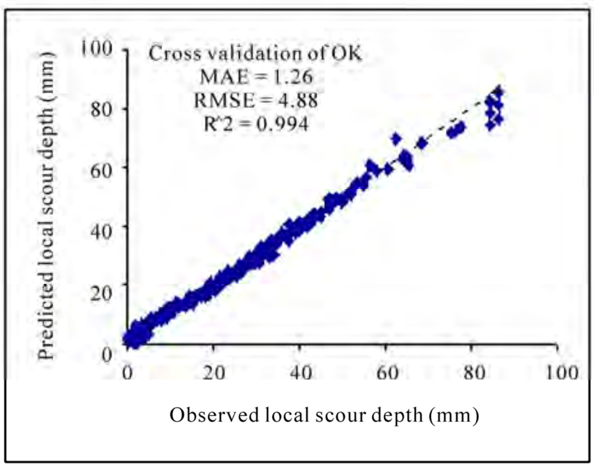
(a)



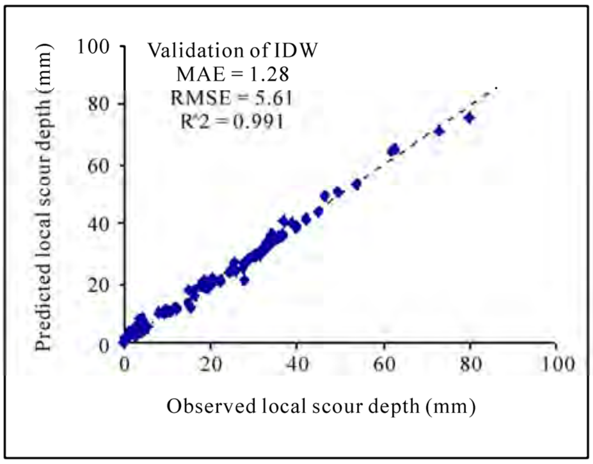
(b)



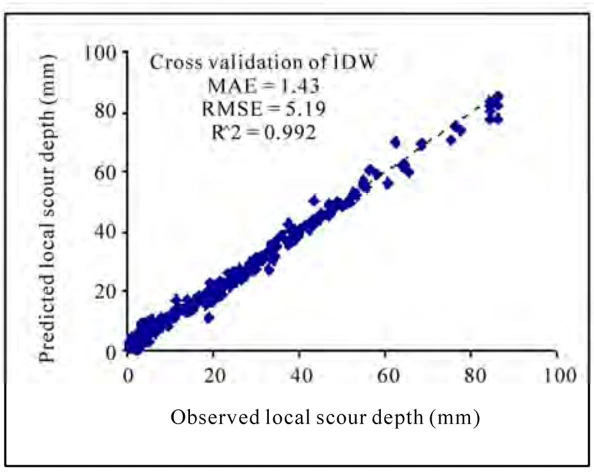
(c)



(d)

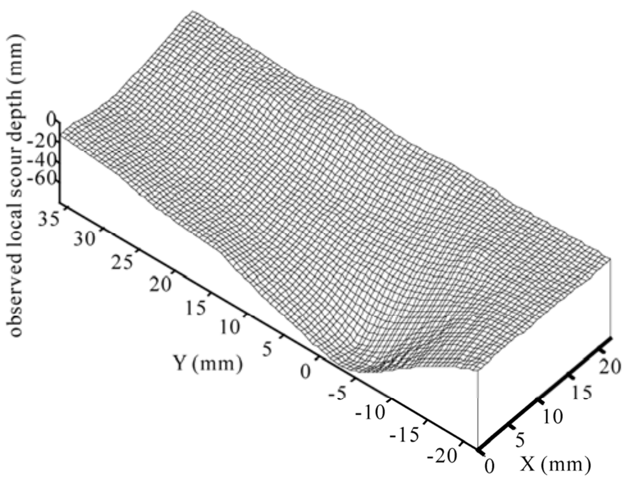


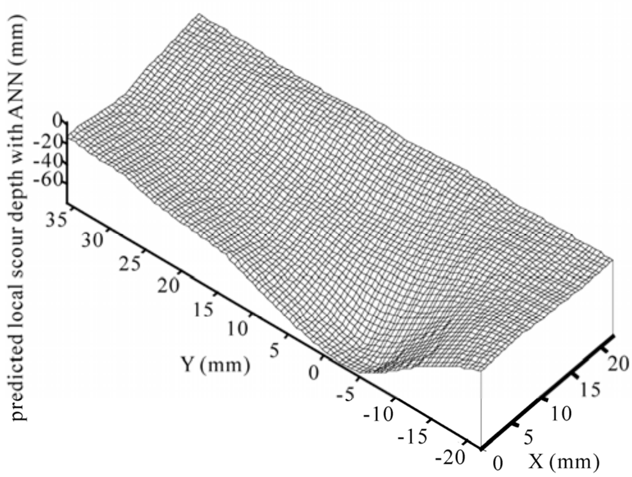
(e)



(f)

Figure 13. Performance of ANN, OK and IDW for the fifth test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.





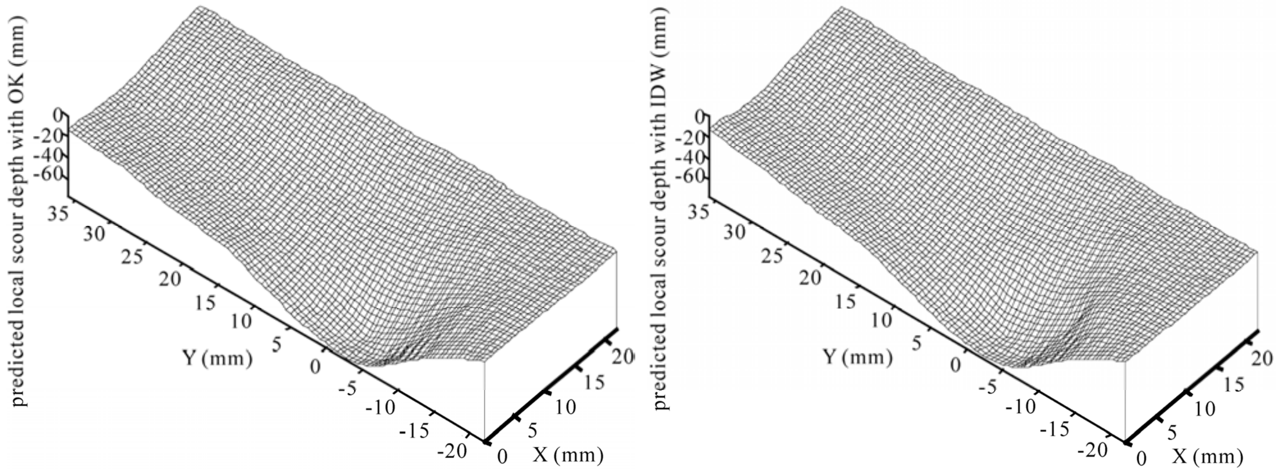
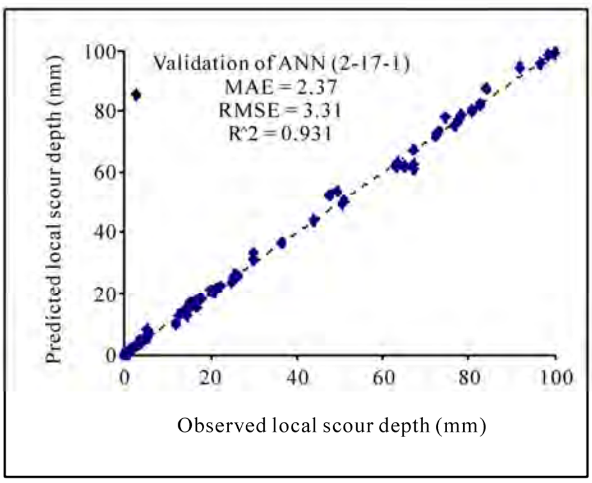
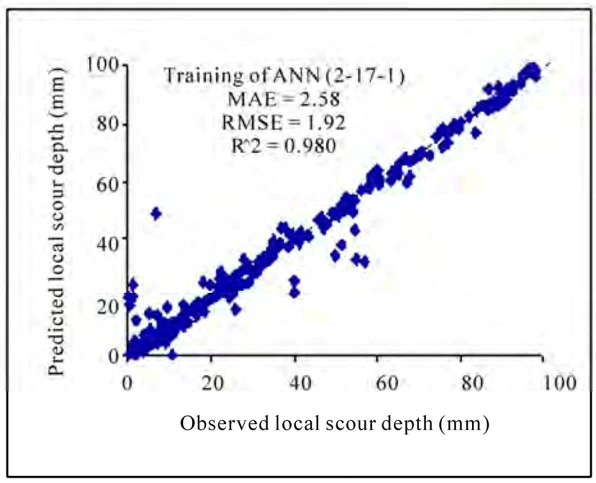


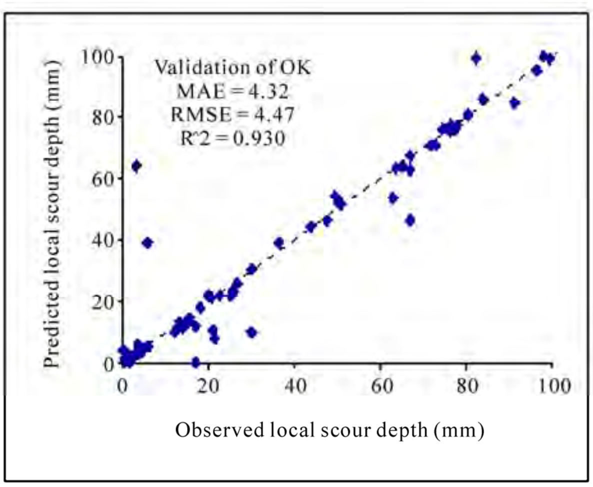
Figure 14. Interpolated maps of observed data, ANN, OK and IDW test for fifth Test case.



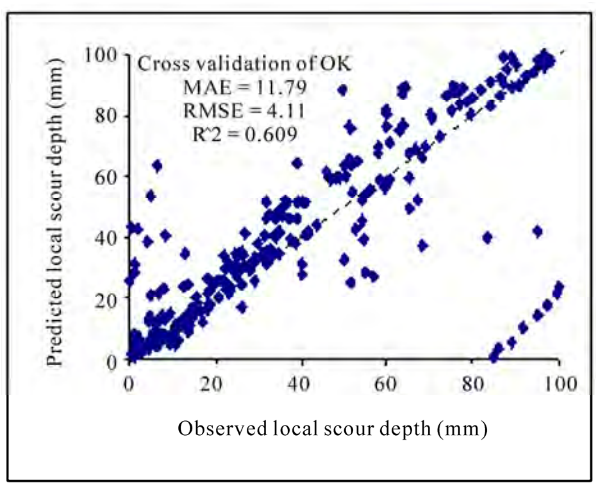
(a)



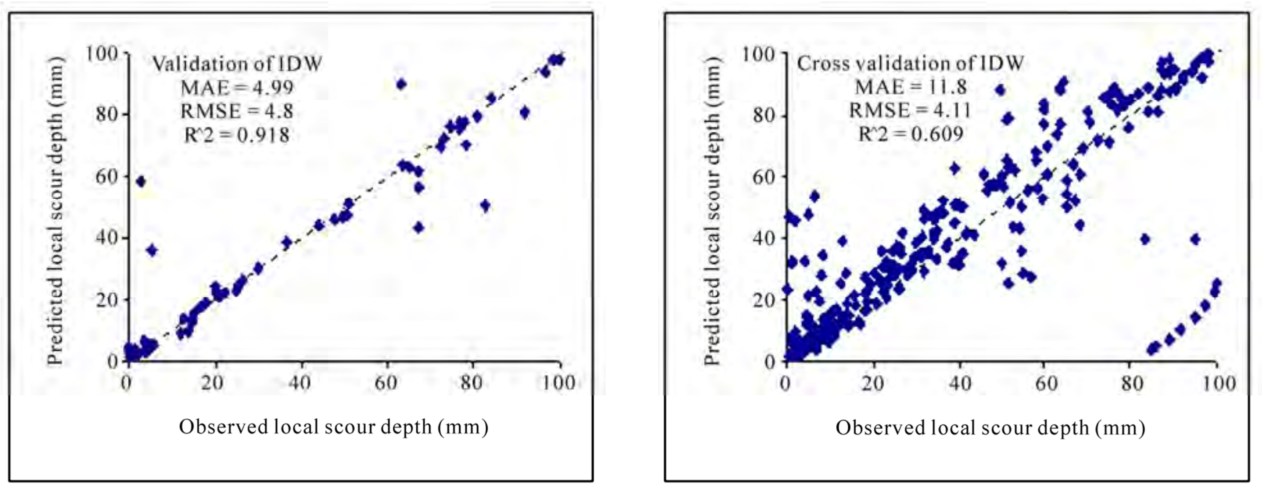
(b)



(c)



(d)



(e) (f)

Figure 15. Performance of ANN, OK and IDW for the sixth test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.

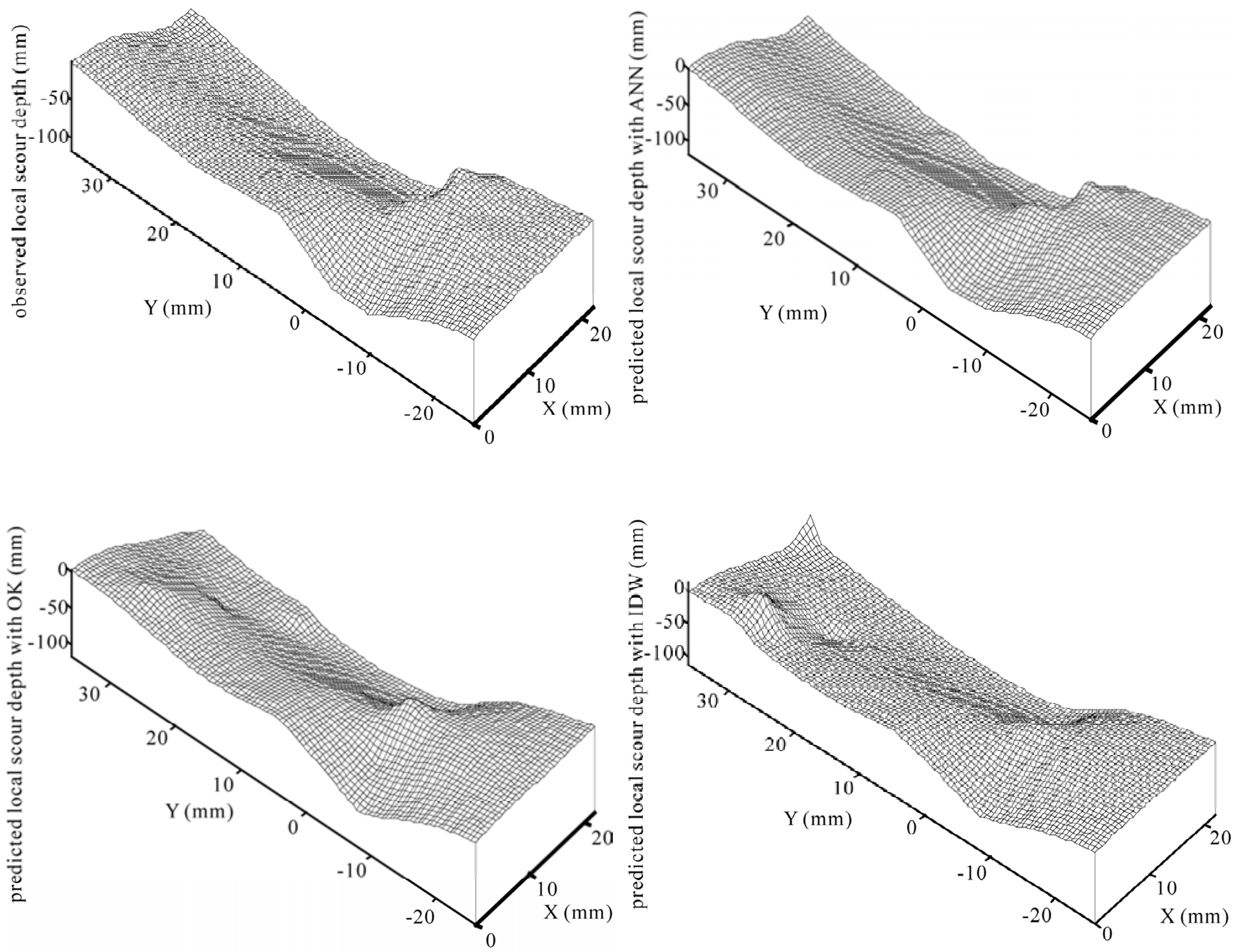


Figure 16. Interpolated maps of observed data, ANN, OK and IDW test for sixth Test case.

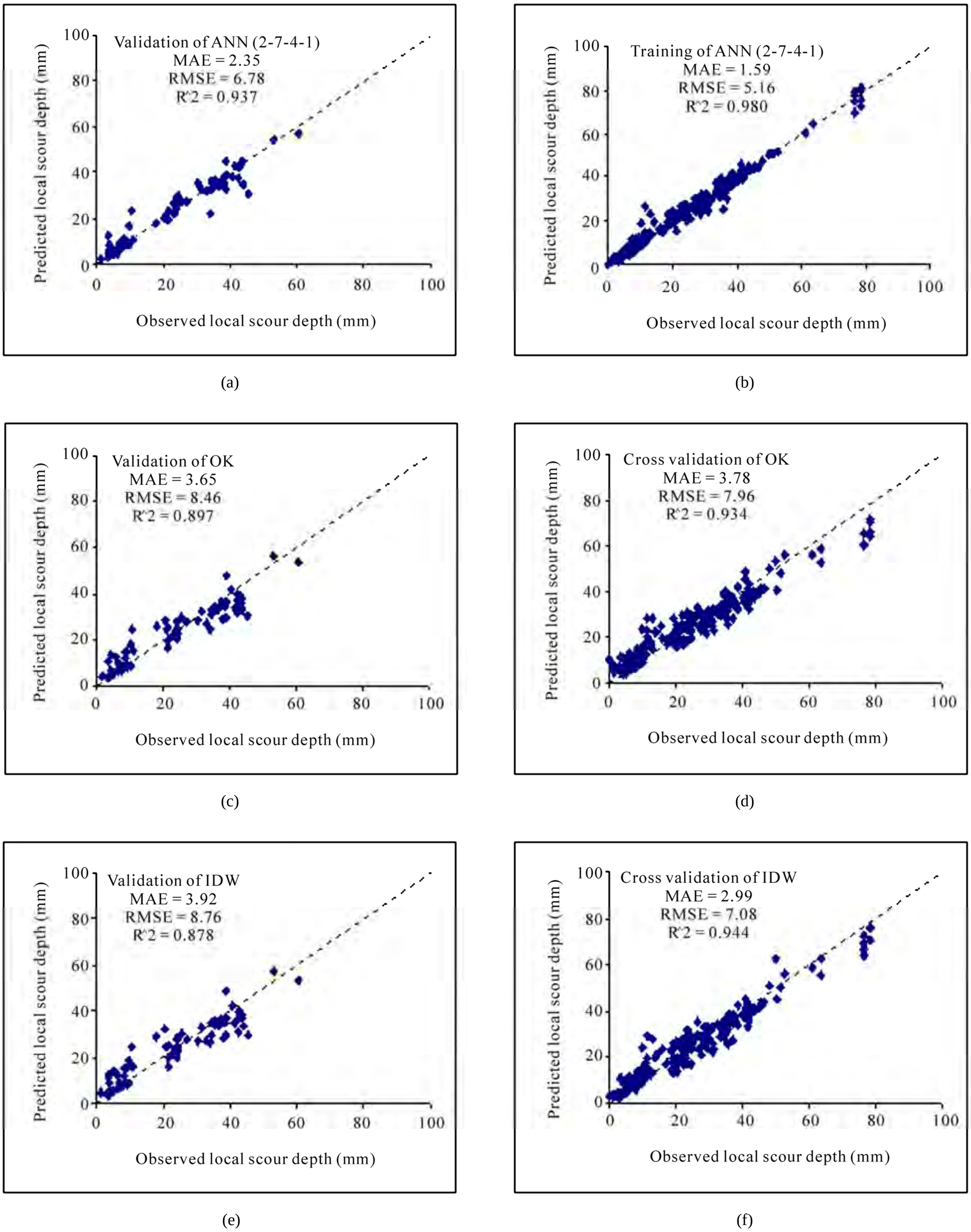


Figure 17. Performance of ANN, OK and IDW for the seventh test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.

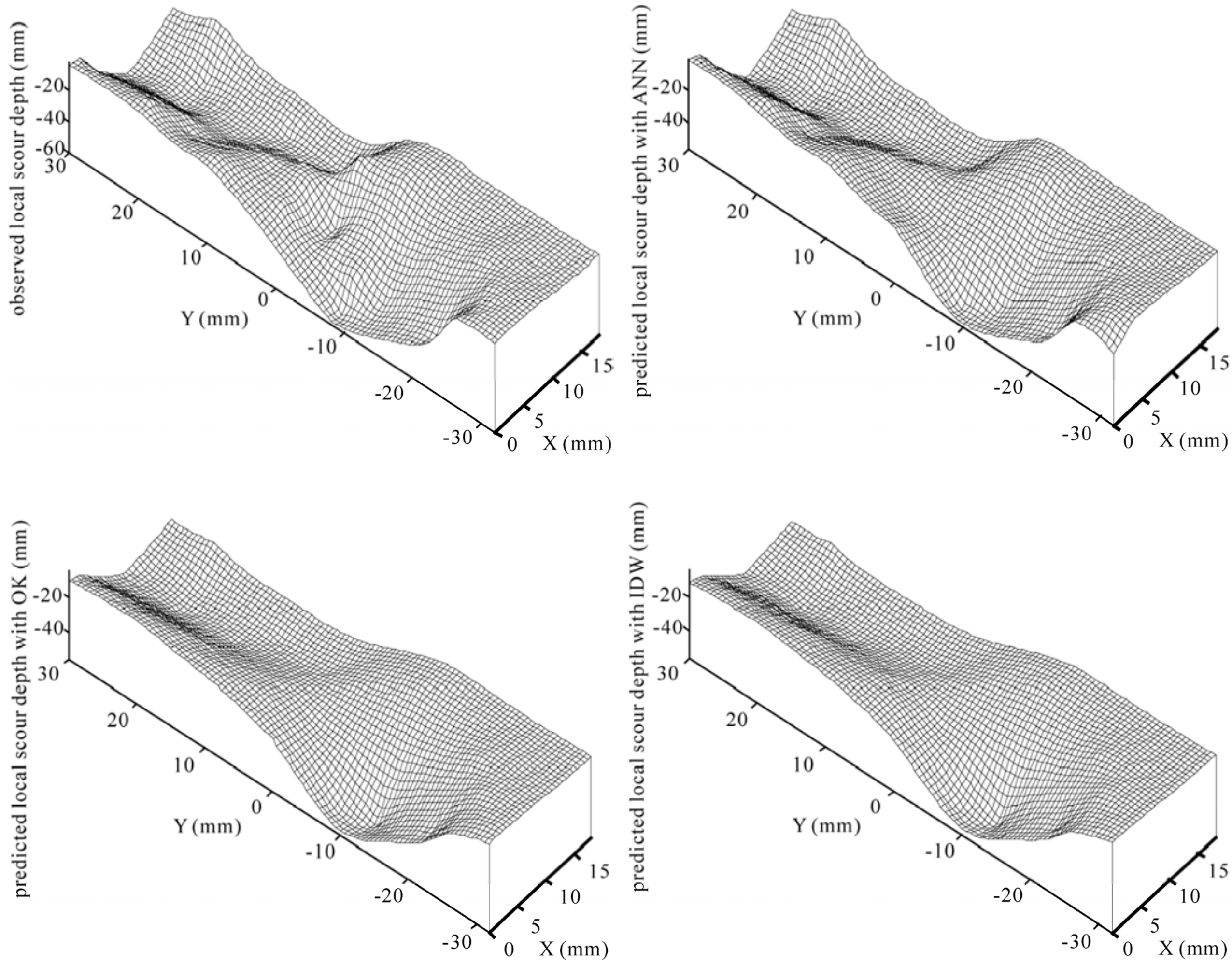
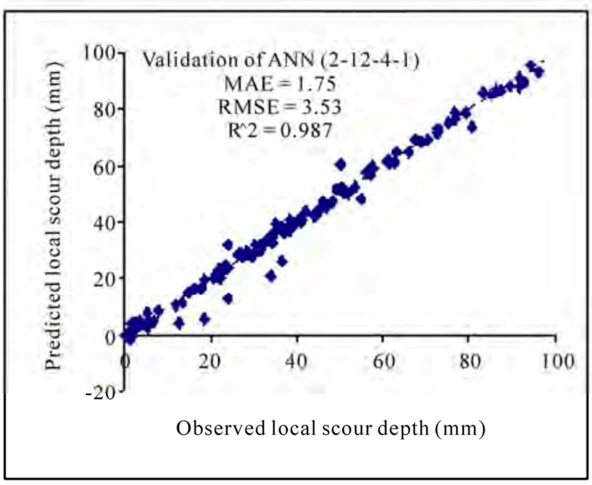
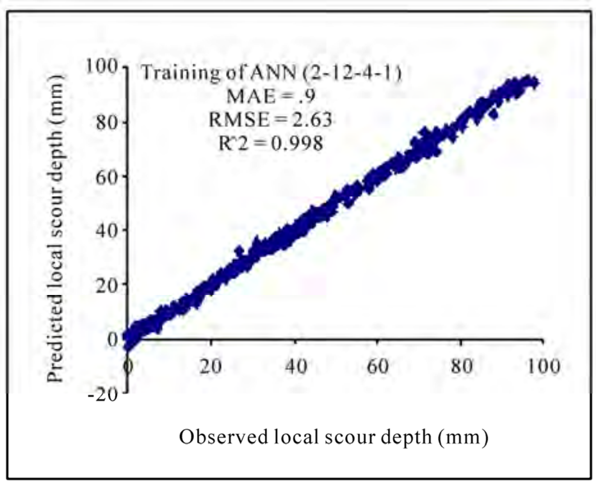


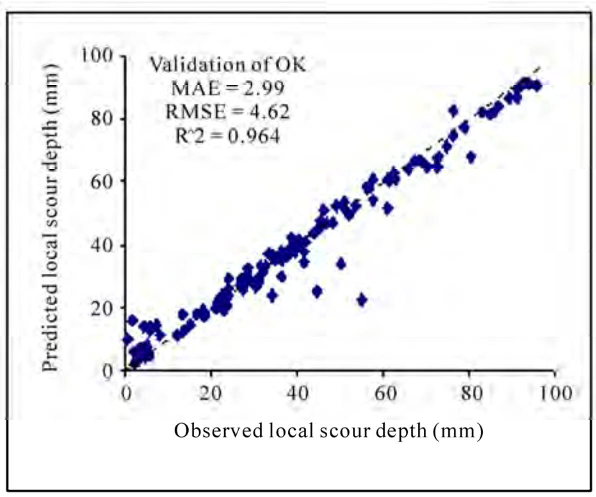
Figure 18. Interpolated maps of observed data, ANN, OK and IDW test for seventh Test case.



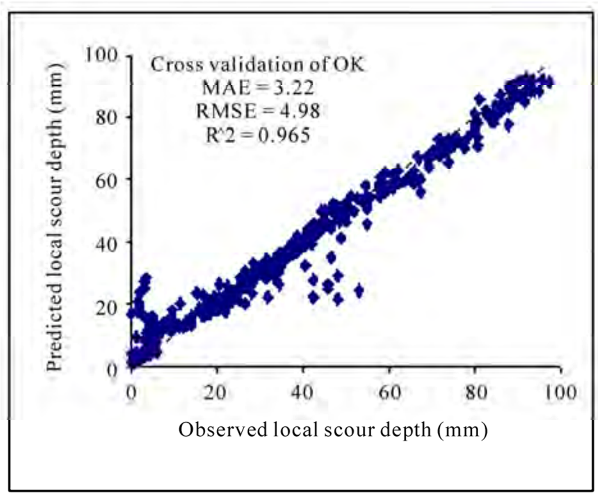
(a)



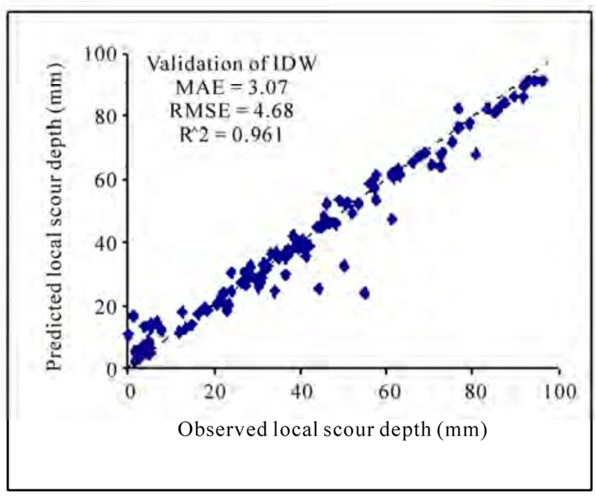
(b)



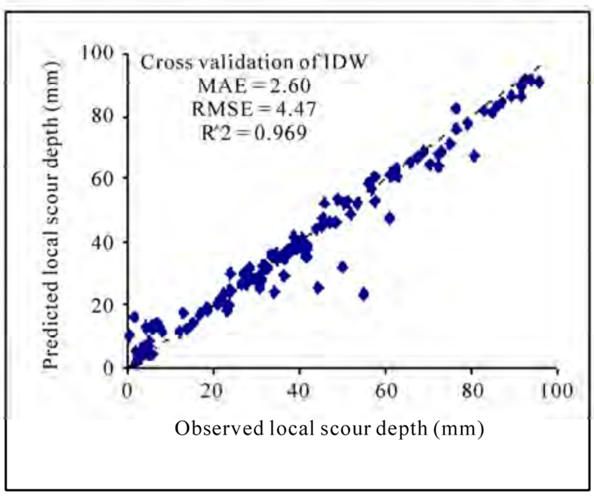
(c)



(d)



(e)



(f)

Figure 19. Performance of ANN, OK and IDW for the eighth test case: a) ANN Validation; b) ANN Training; c) OK Validation; d) OK Cross Validation; e) IDW Validation; f) IDW Cross Validation.

however the diameter of bed sill was equal to 1.2 times the channel width.

From ANN prediction, the optimal architecture was determined by varying the number of hidden neurons (from 1 to 20), and the best structure was selected. It was found that best structure has only one hidden layer and its architecture has the configuration of 2-17-1.

Figure 15 shows training and validation results, respectively. Again, the ANN model performs much better in training and validation. In OK and IDW the accuracy of training is less than validation, but OK showed the best results after ANN. An interpolated map of ANN (**Figure 16**) conforms to the interpolated map of observed local scour depth but the OK and IDW interpolated maps are not matched to the observed interpolated map.

3.8. Seventh Test Case

This seventh test case was similar to the third case, however the diameter of bed sill was equal to 1.2 times the flume width.

From ANN prediction the best structure was selected. It was found that the most accurate results involved use of configuration 2-7-4-1.

Figure 17 depict training and validation results separately for ANN, OK and IDW, respectively, for the seventh data set. When these methods are compared, it was shown that ANN had the best accuracy and IDW had a

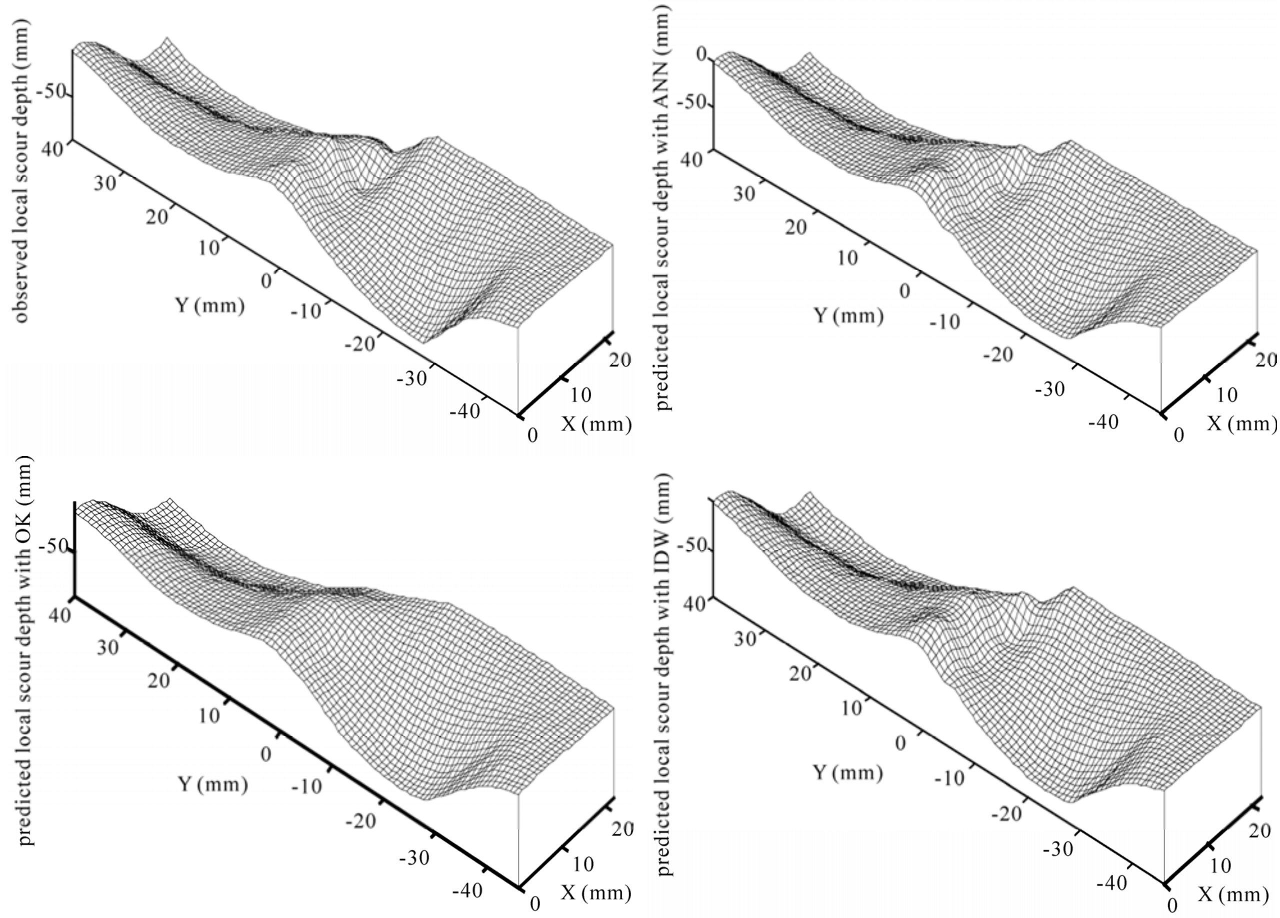


Figure 20. Interpolated maps of observed data, ANN, OK and IDW test for eighth Test case.

low accuracy. Comparisons between these three interpolated maps show that the interpolated map of ANN is very similar to the interpolated map of observed local scour depth (see **Figure 18**).

3.9. Eighth Test Case

In the eighth test case, the bed sill was set in the flow direction with a distance of 25cm between the bridge pier and bed sill. From ANN prediction the best structure was selected. It was found that the most accurate results involved the use of a configuration 2-12-4-1. In the estimate from this test case, like the last seven estimates, ANN has better results when compared with the other methods (**Figure 19**) and the difference in accuracy between OK and IDW isn't significant, as confirmed by the interpolated maps (**Figure 20**).

4. Appropriate Methods for Prediction of Local Scour Depth

In order to identify the most accurate of the three presented methods, **Table 1** represents the results of the research for all conditions. For all conditions, the ANN has the best accuracy in training and validation. ANN performance shows significant preciseness under every condition for the prediction of local scour depth. Only in the fifth test case, the difference in accuracy was very close for all methods. Performance of OK reveals good accuracy, but it has lower precision when compared with the ANN performance. In OK and IDW, accuracy of training is almost the same but OK has better precision in validation performance. Consequently, IDW shows lower performance over all conditions.

5. Conclusions

This paper outlines the application of artificial neural network (ANN), namely the multi-layer perceptron, the ordinary kriging (OK) and the inverse distance weighting (IDW) models in the estimation of local scour depth around bridge piers where bed sills have been installed.

The results of this study showed that the ANN model gives more accurate local scour depth predictions than the existing methods. As such, it is recommended that the ANN model be used for local scour depth predictions instead of the kriging and inverse distance weighting models.

The ANN with two hidden layers was selected as the optimum network to predict local scour depth, whereas the network that includes one hidden layer and 17 hidden nodes within that layer was the best model to predict local scour depth as it is shown in the sixth test case with D/W = 1.2 and r = 5 cm. Also three layers was found the best model to predict local scour depth for test case with no sill as it is shown in the fifth test case.

REFERENCES

[1] D. S. Jeng, S. M. Bateni and E. Lockett, "Neural Network Assessment for Scour Depth Around Bridge," Department of Civil Engineering, The University of Sydney, Sydney, NSW, *Australia, Environmental Fluids/Wind Group, Research Report No R*855, 2006.

[2] N. E Yankielun and L. Zabilansky, "Laboratory Investigation of Time-Domain Reflectometry System for Monitoring Bridge Scour," *Journal of Hydraulic Engineering, ASCE*, Vol. 125, No. 12, 1999, pp. 1279-1284.

[3] P. A. Johnson, "Comparison of Pier-Scour Equations Using Field Data," *Journal of Hydraulic Engineering, ASCE*, Vol. 121, No. 8, 1995, pp. 626-629.

[4] R. Trent, N. Gagarain and J. Rhodes, "Estimating Pier Scour with Artificial Neural Networks," *Proceedings of ASCE Conference on Hydraulic Engineering*, San Francisco, CA, 1993a, pp. 1043-1048.

[5] R. Trent, A. Molinas and N. Gagarain, "An Artificial Neural Networks for Computing Sediment Estimation of Scour Below Spillways Using Neural Networks," *Proceedings of ASCE Conference on Hydraulic Engineering*, San Francisco, CA, 1993b, pp. 1049-1054.

[6] S. Choi and S. Cheong, **"**Prediction of Local Scour around Bridge Piers Using Artificial Neural Networks," *Journal of the American Water Resources Association*, Vol. 42, No 2, 2006, pp. 487-494.

[7] J. B. Butcher, "Co-kriging to Incorporate Screening Data: Hudson River Sediment," *Journal of the American Water Resources Association*, Vol. 32, No. 2, 1996, pp. 349-356.

[8] B. Biglary and T. W. Sturm, "Numerical Modeling of Flow around Bridge Abutments in Compound Channel," *Journal of Hydraulic Engineering, ASCE*, Vol. 124, 1998, pp. 156-164.

[9] S. L. Liriano and R. A. Day, "Prediction of Scour Depth at Culvert Outlets Using Neural Networks," *Journal of Hydroinformatics*, Vol. 3, 2001, pp. 231-238.

[10] A. R. Kambekar and M. C. Deo, "Estimation of Group Pile Scour Using Neural Networks," *Applied Ocean Research*, Vol. 25, No. 4, 2003, pp. 225-234.

[11] S. M. Bateni, S. M. Borghei and D. S. Jeng, "Neural Network and Neuro-Fuzzy Assessments for Scour Depth around Bridge Piers," *Engineering Applications of Artificial Intelligence*, Vol. 20, No. 3, 2007a, pp. 401-414.

[12] T. L. Lee, D. S. Jeng, G. H. Zhang and I. H. Hong, "Neural Network Modeling for Estimation of Scour Depth around Bridge Piers," *Journal of hydrodynamic*, Vol. 19, No. 3, 2007, pp. 378-386.

[13] S. M. Betani, D. S. Jeng and B. W. Melville, "Bayesian Neural Networks for Prediction of Equilibrium and Time-Dependent Scour Depth Around Bridge Piers," *Advances in Engineering Software*, Vol. 38, No. 2, 2007b, pp. 102-111.

[14] Mathworks, "Neural Network Toolbox for Use with MATHLAB," *User's Guide*, The MathWorks, Inc., Natick, MA, 2006, pp. 2-1 to 3-36.

[15] Golden Software User's guide, "Surfer Contouring and 3D Surface Mapping for Scientists and Engineers," 8th Edition, Golden Software, Inc., Golden, CO, 2002.

J. Software Engineering & Applications, **2010, 3, 965-971**
doi:10.4236/jsea.2010.310113 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Scientific Research

Program Slicing Based Buffer Overflow Detection*

Yingzhou Zhang[1,2,3], Wei Fu[1,2], Xiaofei Qian[1], Wei Chen[1]

[1]College of Computer, Nanjing University of Posts and Telecomm., Nanjing, China; [2]State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing, China; [3]State key Laboratory Of Networking & Switching Technology, Beijing University of Posts & Telecomm., Beijing, China.
Email: zhangyz@njupt.edu.cn

Received July 26th, 2010; revised August 16th, 2010; accepted 26th, 2010.

ABSTRACT

The development of the information technology has brought threats to human society when it has influenced seriously the global politics, economics and military etc. But among the security of information system, buffer overrun vulnerability is undoubtedly one of the most important and common vulnerabilities. This paper describes a new technology, named program slicing, to detect the buffer overflow leak in security-critical C code. First, we use slicing technology to analyze the variables which may be with vulnerability and extract the expressions which will bring memory overflow. Secondly, we utilize debug technology to get the size of memory applied by the variable and the size of memory used for these code segments (the slicing result) further. Therefore we can judge whether it will overflow according to the analysis above. According to the unique excellence of program slicing performing in the large-scale program's debugging, the method to detect buffer overrun vulnerability described in this paper will reduce the workload greatly and locate the code sentences affected by corresponding variable set quickly, particularly including the potential vulnerability caused by parameter dependence among the subroutines.

Keywords: *Program Slicing, Buffer Overflow, Inter-Procedure Slicing, Debug, System Dependence Graph*

1. Introduction and Related Work

As early as the beginning of 1970s, buffer overflow has been wildly believed that it is caused by the defects of C language's design model. Array and pointer references are not automatically bounds-checked, so programmers must be up to do these checks by themselves. It is more important that many of the string operations supported by the standard C library such as strcpy(), strcat(), sprintf(), gets(), are unsafe actually. They directly copy data with unknown size to the fixed buffer (as shown in **Figure 1**), causing data overwriting in memory, access violation, or execution of malicious code designed by hackers. The data from CERT shows that 55 percent of general injection attacks are buffer overflow attacks [1], and that among the thirty new vulnerabilities happened during April 19 to May 30 in this year, buffer overflow holds 40 percent [2].

Currently, there are some problems to be solved. Tools with static buffer overflow detection based on string matching algorithms maintain a high rate of false alarms. If the functions existed in the program match the vulnerability database carrying by themselves, they give the corresponding reports.

Now, the common detection methods check the bound of arrays. They regard the memory space variable applied as integer range, for example, char e.g. [10], and its range is [1, 10]. When data are copied into it, we must judge whether the buffer overflows. But problems are still existed. First, we are hard to know how many sizes of the buffer have been used. Because we used to call library function to operate the string, and almost all of the library

```
// no buffer overflow
void NoVulFunc(void)
{
char dest[6];
……
strcpy(dest,"Hello");
……
}
```

```
// buffer overflow
void VulFunc(void)
{
char dest[6];
……
strcpy(dest,"HelloWorld");
……
}
```

Figure 1. An example of a buffer overflow.

*This work was supported in part by the Natural Science Foundation of China (60703086, 60873231, 60873049, 60973046), Jiangsu Natural Science Foundation of China (BK2009426).

function's source code is private or existed in DLLs. Second, these tools are hard to deal with a program with multiple procedures. Lastly, tracking every variable's buffer is almost impossible in a large project.

To solve these problems, many researchers have presented methods and tools for detecting buffer overflows.

1) *The static detection methods based on string matching*. Through string matching, the UNIX's tool, *grep*, can be used to find the unsafe library function call. The tool *RATS* [3,4], developed by Secure Software inc., also studies the unsafe function call in the source code. It matches with the vulnerability database, and then gives the description of the vulnerabilities. Those tools as mentioned above can hardly analyze the semantics and grammar of a program, thus they have many limitations and high false positive.

2) *The detection method based constraint analysis. Splint* [5], developed by university of Virginia, is a static analysis tool used to detect the vulnerabilities in the C program. It adds information of constraint to the source code, and then makes lexical, grammar and semantic analysis for the source. It judges the leaks that may happen in a program and gives the related instructions. From the workflow of the Splint, its limitation is that it requests analysts to be known the source well, and to add notations to the interesting variables and functions.

3) *The dynamic detection methods*. The dynamic detection tool of buffer overflows first inserts some detecting codes in the place where it may happen buffer overflow, then compiles and executes the codes after inserting, and finally, detects whether overflow happens during the execution. But the dynamic detection has defects of efficiency and coverage.

Against the problems described above, this paper proposes a novel method of program analysis, program slicing [6-8], to detect the buffer overrun.

2. Program Slicing

Program slicing [6], originally introduced by Weiser, has been widely used in maintenance of software, program debugging, software testing, code analysis, reverse engineering and so on. For example, during the program debugging, we hope to allocate the codes causing the error. But perhaps the program is too large to find by our hands. If we apply program slicing, we can exclude the codes that have nothing to do with the error, then allocate the error in a smaller range.

In general, program slicing technology has experience a lot: from static slicing to dynamic, from forward slicing to backward, from single procedure to multiple, from non-distributed slicing to distributed, etc.

The slice of a program with respect to program point p and variable x consists of all statements and predicates of the program that might affect the value of x at point p (see **Figure 2**). This concept, originally discussed by Mark Weiser, can be used to isolate individual computation threads within a program. In Weiser's terminology, a slicing criterion is a pair <p,v>, where p is a program point and v is a subset of the program's variable.

With the expansion of the program scale, it is inevitable that program contains multiple procedures. So it appears more important to study inter-procedural slicing. Regarding inter-procedural slicing as a question of graph's reachability, S.Horwitz *et al.* [8] introduce system dependence graph (SDG) to represent the program's dependence graph (PDG).

PDG is a directed graph connected by different kinds of vertexes and some edges (e.g. **Figure 3**). And the vertexes include function's entrance node, declaration node, assignation node, control predicate node, function's call site node, parameter node, the FINAL_USE node of every variable, etc. The edges include control dependence edge of program's circuit, parameter-in and parameter edges generated by call and other data dependence edges. If the definition of variable x in node n is a reachable definition of node m, node m is *data dependent* on n. Control dependents only exist in condition expression and inside the loop expression.

SDG (e.g. **Figure 4**) composes of procedure dependence graph, which is connected by edges that represent direct dependences between a call site and the called procedure and edges that represent transitive dependences due to calls.

```
// a simple example
1    void EgForSlicing ()
2    {
3        int a = 0;
4        int b = 0;
5        int c = 5;
6        if(c > 10)
7            a++;
8        else
9            b++;
10   }
```

```
Slice for variable c in 6th expression
1    viod EgForSlicing ()
2    {
5        int c = 5;
10   }
```

```
Slice for variable a in 7th expression
1    viod EgForSlicing ()
2    {
3        int a = 0;
6        if(c > 10)
7            a++;
10   }
```

```
Slice for variable b in 9th expression
1    viod EgForSlicing ()
2    {
4        int b = 0;
6        if(c > 10)
8        else
9            b++;
10   }
```

Figure 2. A sample of program slicing.

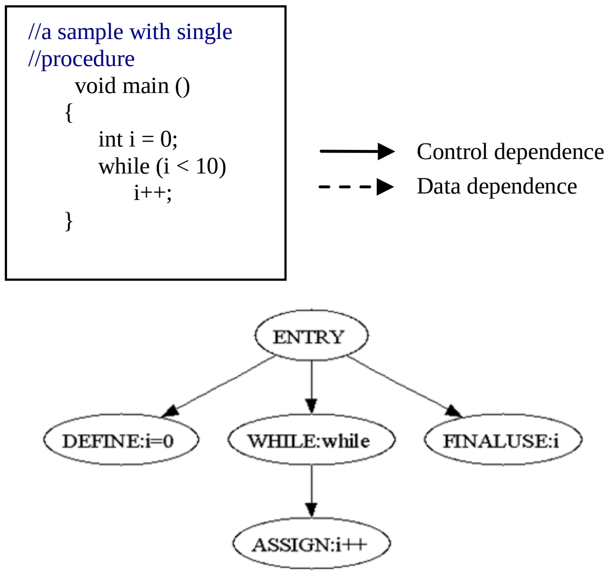


Figure 3. A program and its PDG.

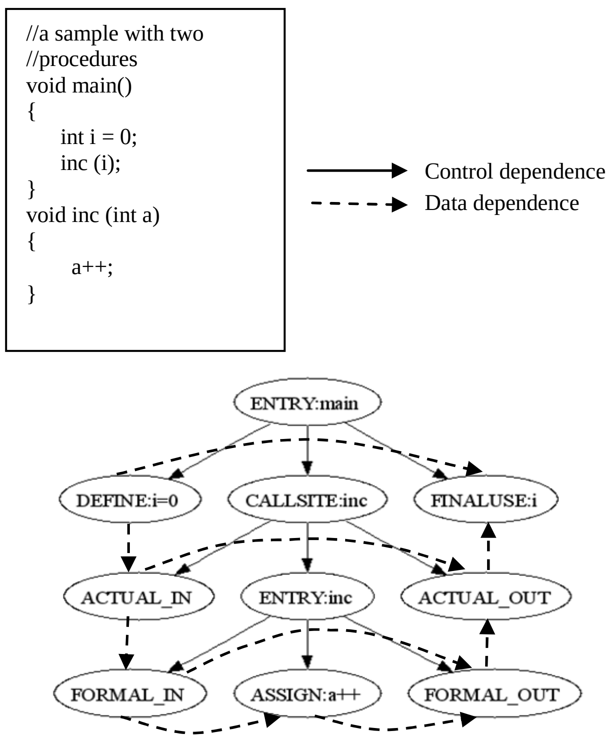


Figure 4. A sample program and its SDG.

For every call site, we use four sorts of vertices to denote the parameter passing: on the calling side, information transfer is represented by a set of vertices called actual-in and actual-out vertices. These vertices, which are control dependent on the call-site vertex (see **Figure 4**), represent assignment statements that copy the values of the actual parameters to the call temporaries and from the return temporaries, respectively. Similarly, information transfer in the called procedure is represented by a set of vertices called formal-in and formal-out vertices. These vertices, which are control dependent on the procedure's entry vertex, represent assignment statements that copy the values of the formal parameters from the call temporaries and to the return temporaries, respectively.

3. An Algorithm of Buffer Overflow Detection

In this section, we will show in detail an algorithm of detecting buffer overflow through program slicing. The algorithm includes four steps as follows.

Step 1. *Constructing PDGs*

Through the lexical and syntax analysis (with Flex and BYACC) of the program to be detected, we first construct the vertices in PDGs. There are usually seven kinds of vertexes as follows.

1) The beginning vertex of compound statements which includes function, if-else, switch statements and so on;

2) The end vertex of compound statements;

3) Call-site vertex;

4) Actual-in and actual-out vertex;

5) Formal-in and formal-out vertex;

6) FinalUse vertex;

7) Others vertex such as ordinary definition statements (e. g.: *int* i), predicate vertexes (e. g.: the judgment part of a *if* statement) and jump statements (e. g.: *break*, *goto* and so on).

Then we construct the edges of PDGs by analyzing the program dependences (includes control dependences and data dependences, see **Figure 4**).

Step 2. *Constructing SDG*

According to the PDGs in Step 1, the SDG can be constructed by the following substeps:

1) For each call site, a call edge from the call-site vertex to the corresponding procedure-entry vertex, is added into the PDG related.

2) For each actual-in vertex v at a call site, we add in PDGs a parameter-out edge from v to the corresponding formal-in vertex in the call procedure;

3) For each actual-out vertex v at a call site, we add in PDGs a parameter-out edge to v from the corresponding formal-out vertex in the called procedure;

4) We finally add in PDGs an edge between actual-out vertex to actual-in vertex if they are reachable between the corresponding formal-out and the corresponding formal-in.

Step 3. *Computing program slices by traversing SDG in two phases*

Supposing that we want to computer the inter-procedure slice of the variables in vertex n.

Phase 1: Starting from the vertex n, we travel reversely the SDG through the control dependence edges and data dependence edges (not including parameter-in edges), and then mark all of reachable vertexes.

Phase 2: Starting from the vertexes marked in phase 1, through the control edges (not including call-site edges) and data dependence edges (not including parameter-out edges), we mark all of reachable vertexes.

The result of the inter-procedure slice is the union set of the marked vertexes in above two phases.

Step 4. *Detecting Buffer Overflow*

For each variable needed to analyze, we can use a data structure to store the information of usage. The data structure includes variables, the flags of overflow, the usage information and the total size of memory applied by a variable. For example, char p [10], the detection model about p is as follows (see **Figure 5**).

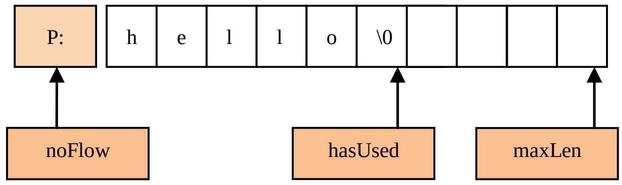


Figure 5. The detection model of buffer overflow.

From the abstract syntax tree obtained by lexical analysis and syntax analysis, we can get each call function. Through the inter-procedure slicing of the parameters of the vulnerable functions, we then can obtain a piece of executable code segment which may affect these parameters. At last, we watch the memory of the variable for judging whether the buffer related is overflow, by setting breakpoints in a debugger at the beginning of the main function and the vulnerable function.

4. Implementation

Through the intermediate representation AST of a program (from the lexical and syntax analysis of the program), we can construct some useful graphs such as PDGs and SDGs. In PDGs, the data structure of nodes and edges are as follows.

```
struct PDGNode {
        CFGType type;
 //type of node, such as "DEFINE", "IF", "WHILE". etc
        StructId    astId;
    //node's ID in AST
        PDGId       father;
    //connect to the control predicate
        PDGId       tBranch, fBranch;
    //the PDGID of true branch or false branch
            PDGId       ifd;
    //post-dominator
        ListId      prevHead;
  //the first node'ID of the father node table
        PDGId       callLink;
  //link to the first function call site
        DepEdgeId cdFirst;
  //the ID of the node's first control dependence edge
        DepEdgeId ddFirst;
  //the ID of the node's first data dependence edge
        };
struct DepEdge {
            int         depSrc;
  //the PDG ID of dependence source
            int         depDest;
  //the PDG ID of dependence destination
            DepEdgeId srcNext;
//the next dependence edge with the same dependence
  //source
            DepEdgeId destNext;
//the next dependence edge with the same dependence
  //destination
        };
```

According to the algorithm in above section, we can construct SDGs from PDGs by adding some edges with the following functions.

1) Adding control dependence edges from call-site to the callee's body, through the function AddEdge (CTRL_EDGE, callSite, entry).

2) Adding data dependence edges from actual-in to formal-in, through the function AddEdge (DATA_EDGE, Actual_in, Formal_in)

3) Adding data dependence edges from formal-out to actual-out, through the function AddEdge (DATA_EDGE, Formal_out, Actual_out)

The function AddEdge inserts dependence edges to the dependence table according to the edge's style. Then we assign an ID of the new edge to the destNext (a member variable of DepEdge). The following codes show in detail the implementation of AddEdge.

```
AddEdge (int kind, int from, int to)
  {
      int lastEdge;
      DepEdge dep (from, to);
      depTable. insert (dep);
      switch (kind)
      {
        case CTRL_EDGE:
          lastEdge = pdgTable [from]. cdFirst;
          break;
        case DATA_EDGE:
          lastEdge = pdgTable [from] .ddFirst;
          break;
        }
      if (lastEdge < 1)
```

```
//if this node has no dependence edge
{
pdgTable [from]. cdFirst = depTable. getCurId();
return;
}
//if the node has dependence edge already
    while ((depTable [lastEdge]. depDest! = to)&&
       depTable [lastEdge]. destNext >= 1)
    lastEdge = depTable [lastEdge]. destNext;
    depTable [lastEdge]. destNext = depTable.getCurId ();
}
```

The dependence between the actual parameter can be formed according to the dependence among the corresponding formal parameters. When we analyze the library function, we suppose that each actual-out is dependent on the actual-in. So we need not go deep into the inner of the library function body. The detail codes of the dependence among formal parameters are as follows.

```
void BuildInterActualParameterDep (int fnId)
{
  //paramNum is the number of the parameter
   for(int i = 0; i < fnTable[fnId]. paramNum; i++)
  //judge whether the corresponding formal-in can
  //reach the corresponding formal-out
   if (IsReachable (formal-in, formal-out))
   //Add edge between the corresponding actual-in
  //and the corresponding actual-out;
   AddEdge (DATA_EDGE, actual-in, actual-out);
}
```

According to the algorithm described in section 3, we need to traverse the SDG in two phases. Because parameter-out edges are not followed, the traversal in phase 1 does not descend into procedures. But the effects of such procedure are not ignored. The presence of transitive flow dependence edges from actual-in to actual-out vertices permits the discovery of vertices that can reach the vertex you want to slice through a procedure call, although the graph traversal does not actually descend into the called procedure. In phase 2, because call edges and parameter in edges are not followed, the traversal does not ascend into the calling procedure; the transitive flow dependence edges from actual-in to actual-out vertices make such ascents unnecessary. So we can solve the call-context problem (as shown in the following codes).

```
void InterProSlice (SDGs, PDGId vulPdgNode)
 {
 //phase 1: traverse in calling procedure
 ReachingNode (s, vulPdgNode, {parameter_out});
 //phase 2: traverse in called procedure
 //vSet is a set marked in phase 1;
 ReachingNode(s, vSet, {parameter_in, call});
 }
void ReachingNode(SDG s, vSet, kind)
    //vSet is the set of vertex
    //kind is type of the vertex
{
  stack<PDGId> nodeStack;
  push the vertices that exist in the vSet to the stack;
  while (!nodeStack .IsEmpty())
   {
      pop a vertex v from the stack;
      mark the vertex v;
      while (if v's depTable is not empty)
      {
    push the vertices existed in the depTable to the stack;
      }
    }
}
```

After obtaining the result of the slicing, we set breakpoint at the beginning of the main function and at the place of the vulnerable function, then call the debugger (for example: the debugger embedded in Microsoft Visual Studio or the Zeta debugger [9]) to execute by step until the end. The implementation is as follows:

```
void BufferOverDetect (int start, int end, int vulPos, char * fileName)
    {
       char buf [10];
       ZD_LoadProgram (fileName);
       ZD_SetBreakPoint (start,true);
       ZD_SetBreakPoint (vulPos,true);
       ZD_RunTo (start);
       While (start <= end)
    {
       ZD_RunTo (++start);
    }
  /*the function below is used to read a byte of
    content starting from the memory address of
    add to the buf. The add is passed as follows: if
    you want to check whether vul (vul is defined
    like this: char vul [10]) will be overflowed, we
    just need to watch the content of the vul [10],
    and the add is & vul [10].*/
    ZD_Read (add, 1, buf);
}
```

5. A Sample

In this section, we will show based on program slicing a sample (see **Figure 6**), where the variable of vulBuf will be overflowed.

Due to the restriction of space, parts of the vertices in **Figure 6** have been abbreviated shown as follows.

D:*noR* = *DEFINE*:*noRelated*;
D:*noRBuf* = *DEFINE*:*noRelatedBu*;
D:*vulBuf* = *DEFINE*:*vulBuf*;
A:*noR++* = *ASSIGN*:*noRelated++*;

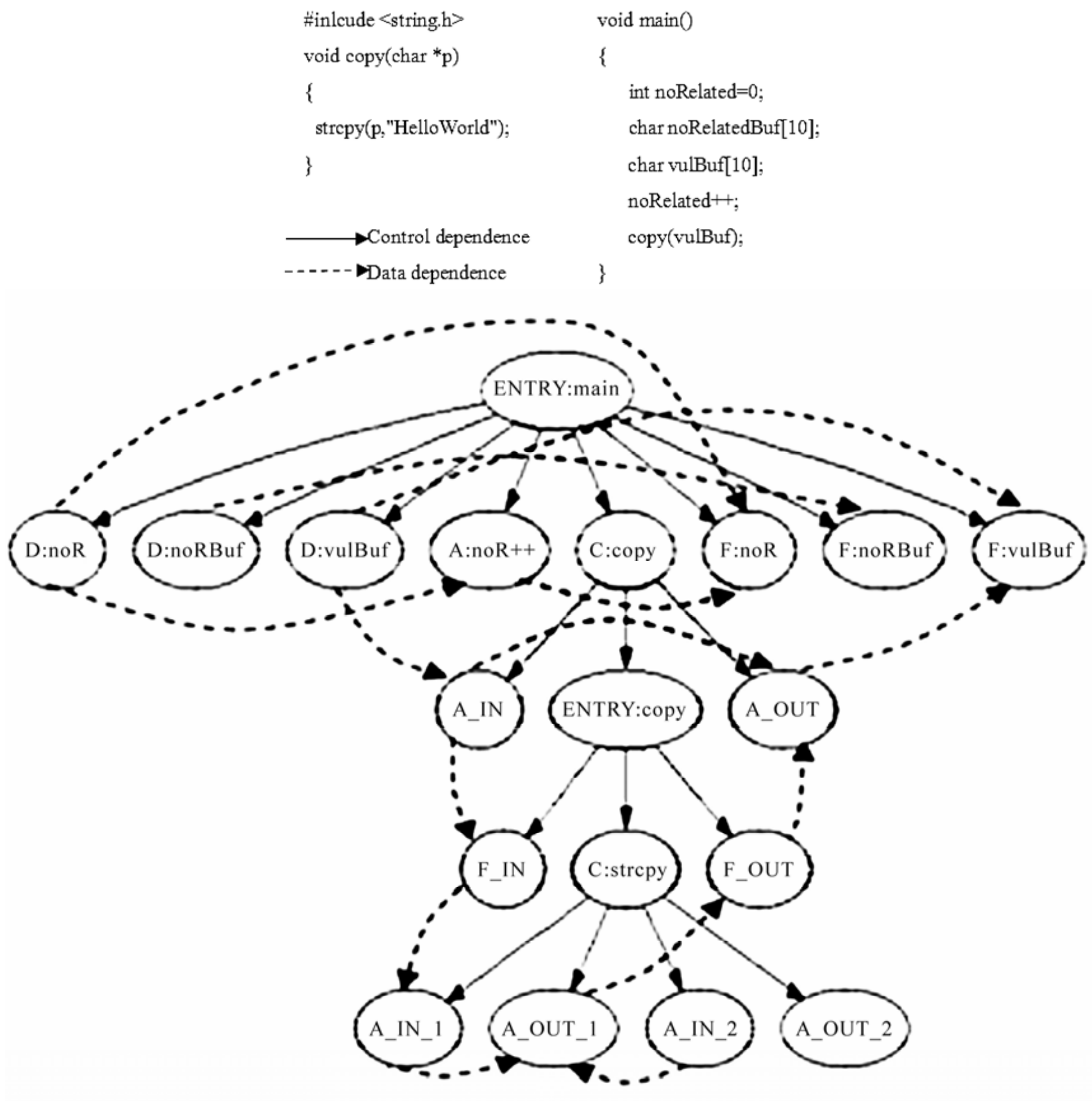


Figure 6. A sample of buffer overflow and its SDG.

C:*copy* = *CALL_SITE*:*copy*;
F:*noR* = *FINALUSE*:*noRelated*;
F:*noRBuf* = *FINALUSE*:*noRelatedBuf*;
F:*vulBuf* = *FINALUSE*:*vulBuf*;
A_IN = *ACTUAL_IN*;
A_OUT = *ACTUAL_OUT*;
F_IN = *FORMAL_IN*;
F_OUT = *FORMAL_OUT*;
*A_IN*_1 = *ACTUAL_IN*_1;
*A_OUT*_1 = *ACTUAL_OUT*_1;
*A_IN*_2 = *ACTUAL_IN*_2;
*A_OUT*_2 = *ACTUAL_OUT*_2;

After constructing the SDG, we start to find the vulnerable function call with the matching. Then we can find the strcpy in the copy may cause overflow. So we make inter-procedure slicing for variable p, and obtain the result showed in **Figure 7**.

Then we call the debugger, and set breakpoint at the beginning of the main function and at the place of the strcpy, by starting to execute by step. At last, we will find vulBuf[10] equals character of NULL (see **Figure 8**). This shows that vulBuf has been overflowed.

After debugging, we will give the corresponding report.

There are some advantages in our detection tool based on program slicing. First, this tool improves the accuracy greatly compared with the ITS4 and RATS which use string matching to detect the buffer overflows. Second, through program slicing, we can get rid of the useless codes. Compared with the detecting tool that sets a constraint for each variable and watches its value, our tool can reduce the variables needed to watch, improve the


```
# include <string.h>
void copy(char *p)
{
        strcpy (p,"HelloWorld");
}
void main()
{
      char vulBuf [10];
      copy (vulBuf);
}
```

Figure 7. The slice result of the sample in Figure 6.

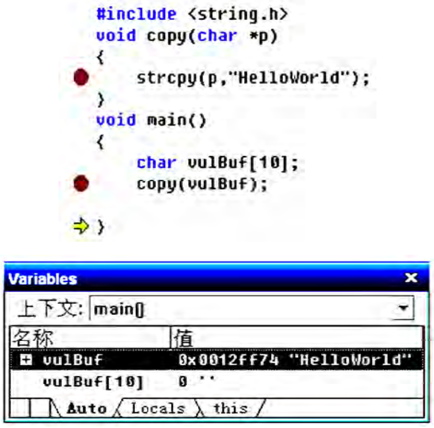


Figure 8. The result of a detection debugger.

performance. Third, compared with the Splint, our tool needs not to insert information of constraint to the source. So the analyst needs not learn the program a lot.

6. Conclusions

This paper introduces inter-procedure slicing to solve the problem of detecting buffer overflow. Compared with the other methods, this method has the high performance and excellent precision. But our tool is only fit for the programs coded by C, and it is still blindness in the object-oriented languages. So eliminating the blindness will be our further work.

REFERENCES

[1] "CERT/CC.Vulnerability notes by metric," http://www.kb.cert.org/vuls/bymetric?open&start=1&count=20

[2] "US-CERT Recently Published Vulnerability Notes," http://www.kb.cert.org/vuls/atomfeed?OpenView&start=1&count=30

[3] "Secure Software, Rough Auditing Tool for Security, (RATS)," Secure Software Inc, http://www.secure software.com

[4] J. Viega, J. T. Bloch, T. Kohno and G. McGraw, "ITS4: a static vulnerability scanner for c and c++ code," *Annual Computer Security Applications Conference*, Hawaii, 2000, pp. 257-267.

[5] D. Evans and D. Larochelle, "Improving Security Using Extensible Lightweight Static Analysis," *IEEE Software*, Vol. 19, No. 1, 2002, pp. 42-51.

[6] M. Weiser, "Program Slicing," *The IEEE Transactions on Software Engineering*, Vol. 10, No. 4, 1984, pp. 352-357.

[7] Y.Z. Zhang and B. W. Xu, "A Novel Formal Approach to Program Slicing," *Science in China, Series E, Information Sciences*, Vol. 38, No. 2, 2008, pp. 161-176.

[8] S. Horwitz, T. Reps and D. Binkley, "Interprocedural Slicing Using Dependence Graphs," *ACM Transaction on Programming Languages and Systems*, Vol. 12, No. 1, pp. 26-60.

[9] "Zeta Debugger," http://www.fyzor.com/debugger/index.htm

J. Software Engineering & Applications, 2010, 3, 972-982
doi:10.4236/jsea.2010.310114 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Software Engineering Principles: Do They Meet Engineering Criteria?

Kenza Meridji, Alain Abran

École de Technologie Supérieure (ÉTS)—University of Québec, Montréal, Québec, Canada.
Email: kenza.meridji.1@ens.etsmtl.ca, alain.abran@etsmtl.ca

Received July 31st, 2010; revised August 25th, 2010; accepted August 31st, 2010.

ABSTRACT

As a discipline, software engineering is not as mature as other engineering disciplines, and it still lacks consensus on a well-recognized set of fundamental principles. A 2006 *analysis surveyed and analyzed* 308 *separate proposals for principles of software engineering, of which only thirty-four met the criteria to be recognized as such. This paper reports on a further analysis of these thirty-four candidate principles using two sets of engineering criteria derived from*: *A*) *the engineering categories of knowledge defined by Vincenti in his analysis of engineering foundations*; *and B*) *the joint IEEE and ACM software engineering curriculum. The outcome of this analysis is a proposed set of nine software engineering principles that conform to engineering criteria.*

Keywords**:** *Engineering Criteria*, *Software Engineering Principles*, *Vincenti*, *Engineering Verification Criteria*

1. Introduction

Software engineering is defined by the IEEE as:

"1) The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software, *i.e.* the application of engineering to software.

2) The study of approaches as in 1" [1].

The intended goals of software engineering are different from those of computer science. In software engineering, artifacts are designed, produced and put into operation, while in computer science the theoretical foundations of information and computation are the object of study. Computer science deals with the investigation and analysis of algorithms and their related problems, in order to enable the computer perform the task [2].

Computer science is the discipline that underlies software engineering, and it is to be expected that the principles for computer science will be different from the principles of software engineering. For example, a principle proposed in computer science, "Use coupling and cohesion" [3], deals with the underlying science, while software engineering principles are more general, like "Apply and use quantitative measurements in decision making" [3].

Of course, software engineering is still an emerging engineering discipline and is not yet as mature as other traditional engineering disciplines such as mechanical and electrical engineering. Much of the research conducted to date in software engineering has focused on developing methods, techniques, and tools, and considerably less on exploring the engineering foundations of software engineering, including identifying the software engineering fundamental principles, or how to apply them in research and practice.

A significant amount of the work carried out to date on software engineering principles has been based on expert opinion, with a few exceptions in which defined research methodologies have been used, such as in [4] and [5], where Delphi rounds are applied to develop an initial consensus among a group of 12 experts on a set of candidate principles for software engineering.

In a 2006, literature survey on this topic, covering the previous 20 years [3], 308 separate proposals for candidate software engineering principles was identified. These were then analyzed against a set of criteria related to the specific concept of a 'principle', following which only 34 were recognized as bona fide 'candidate' fundamental principles (FPs) [3]. However, the research scope of that study did not, include within its research scope an analysis of these candidates from an engineering perspective.

In this paper, we perform that analysis. One of the challenges, of course, is to figure out what criteria should be verified from an engineering perspective, since, in the traditional engineering literature, such criteria are not explicitly described. This paper documents the methodology to make that determination, as well as what we found when we applied them to the set of 34 candidate FPs.

The paper is organized as follows: Section 2 presents related work on software engineering principles. Section 3 presents the analysis methodology selected. Section 4 identifies the software engineering verification criteria. Section 5 describes the application of these criteria. Section 6 presents the analysis results. Section 7 points out the limitations of that work. Section 8 presents our conclusion.

2. Related Work on Software Engineering Principles

The expression "fundamental principle" is composed of two terms. According to the Cambridge University Dictionary, the term "fundamental" means "forming the base, from which everything else originates; more important than anything else", and "principle" means "a basic idea or rule that explains or controls how something happens or works".

From the literature on software engineering principles, the authors of [3] inventoried 308 principles that had been proposed by individuals (for instance [6,8]) or as part of a collaborative effort [9-11]. With the exception of [11] and [3], the authors involved proposed only nominative principles, without including either formal definitions or procedures for implementing them. To verify whether or not each of these was indeed a candidate 'fundamental' principle, in our sense of the terms, a two-step verification process was used in [3] and [3-12]:

1) Identification of seven verification criteria

Five criteria applicable to each proposed principle were derived from [4]:

- A principle is a proposal formulated in a prescriptive way;
- A principle should not be directly associated with, or arise from, a technology, a method, or a technique, or itself be an activity of software engineering;
- The principle should not dictate a compromise (or a proportioning) between two actions or concepts;
- A principle of software engineering should include concepts connected to the engineering discipline;
- It must be possible to test the formulation of a principle in practice, or to check its consequences.

Two additional criteria were identified as applicable across the full set of proposed principles:

- The principles should be independent, e.g. not deduced [6];
- A principle should not contradict another known principle [4].

2) Verification of each of the proposed 308 principles surveyed against these criteria

In [3] it is reported that only 34 of the 308 proposals met the full set of criteria to be recognized as candidate FPs. Table 1 lists them, in alphabetical order [3].

In their paper "Fundamental Principles of Software Engineering–A Journey" [4], the authors identified a set of fundamental principles through a well documented research methodology. They defined the relationships between principles, standards and implemented best practices as illustrated in **Figure 1**.

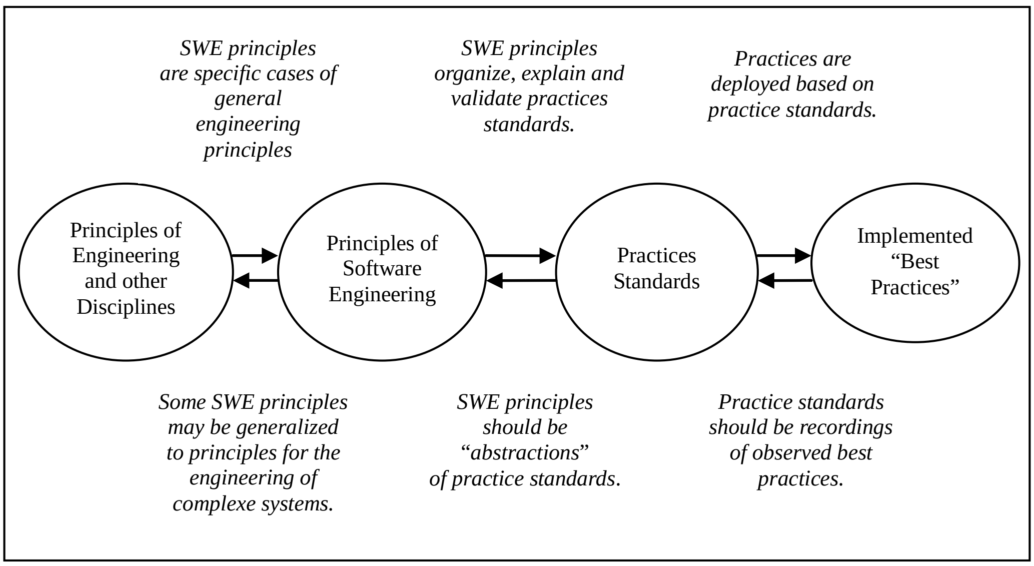


Figure 1. Relationships between principles, standards and practices [4].

This figure illustrates the relationships sought among principles standards and practices. It is believed that a body of fundamental principles has been recorded for some branches of engineering, e.g. (Vincenti, 1990). Most of the engineering branches have a history far longer than that of software engineering and software engineering principles (SWE principles in **Figure 1**) would, in general, be regarded as specializations of these principles. The software engineering principles would play the role of organizing, motivating, explaining, validating the practice standards and implemented practices should be based on those practice standards [4].

Working from the specific toward the general, practice standards would be recordings and idealizations of observed and validated "best practices". The software engineering principles would be abstractions of the practice standards. Furthermore, software engineering principles might be candidates for generalization to the status of general engineering principles, particularly when complexity is a concern [4].

3. Analysis Methodology

The scope of the criteria used in [3] was limited to the concept of 'principles', and did not include the specific features of the engineering concepts themselves. This is the focus of the work reported here: the list of 34 candidate principles in **Table 1** constitutes the input to the analysis process required to verify whether or not they conform to engineering criteria. This will make it possible to narrow the number of principles. More specifically, the research issue addressed in this paper is, which of these 34 candidate FPs conform to engineering criteria?

Of course, engineering criteria are required for this verification and must be available, but no related work could be identified. So, the first challenge was to determine verification criteria from an engineering perspective.

To tackle this issue, it was necessary to study the epistemology of engineering. For that purpose, two sources, Vincenti, the author of the book, What Engineers Know and How They Know It [13], and the joint IEEE-ACM software engineering curriculum [14] were selected:

- Vincenti has identified a number of engineering knowledge types as key to the engineering disciplines and from which engineering criteria can be derived;
- The IEEE and the ACM have documented a set of topics within their joint software engineering curriculum from which engineering criteria can be derived.

The approach designed for identifying relevant criteria and applying them to the set of 34 candidate FPs consists of three phases—see **Figure 2**.

Phase 1: Identification of two sets of verification criteria

This phase consists of the identification of criteria which would be relevant to any engineering discipline. Such criteria could have been taken either as is, when expressly identified and defined, or derived, when documented only implicitly. The inputs to this phase are the two sources of information identified from the related work. The outputs of this phase are the two sets of criteria derived from Vincenti and from the IEEE-ACM joint software engineering curriculum. The criteria identification phase based on Vincenti is summarized in **Figure 3**, which shows its inputs and outputs.

The criteria identification phase based on the IEEE-ACM criteria is summarized in **Figure 4**, which shows its inputs and outputs. This phase is presented in greater detail in Section 4.

Phase 2: Verification execution:

The 34 candidate FPs will be taken as inputs in the second phase and analyzed next with respect to the two sets of engineering criteria identified in Phase 1.

The output will be the FPs that have at least one direct mapping, and those that have only an indirect mapping to either Vincenti or to the IEEE-ACM engineering criteria. This phase is illustrated in **Figure 5** and is presented in greater detail in Section 5.

Phase 3: Analysis and selection:

In Phase 3, the analysis across each set of engineering criteria is performed. This phase identifies the candidate FPs that meet engineering criteria from both sets of criteria and those that do not. For instance, the candidate FPs that meet only the Vincenti criteria (Vincenti, 1990) and the candidate FPs that only meet the IEEE-ACM criteria [12] are then be analyzed to check whether or not they can be identified from the FP that are recognized as engineering FPs. This phase is described in greater detail in Section 6.

4. Phase 1: Identification of Engineering Criteria

4.1. Vincenti

Vincenti [17] studied the epistemology of engineering based on the historical analysis of five case studies in aeronautical engineering covering a roughly fifty-year period and proposed a taxonomy of engineering knowledge. He identified different types of engineering knowledge and classified them into six categories:

1) Fundamental design concepts,
2) Criteria and specifications,
3) Theoretical tools,
4) Quantitative data,
5) Practical considerations, and

Table 1. Inventory of Candidate FPs [3].

No	Proposals–in alphabetical order
1	Align incentives for developer and customer
2	Apply and use quantitative measurements in decision making
3	Build software so that it needs a short user manual
4	Build with and for reuse
5	Define software artifacts rigorously
6	Design for maintenance
7	Determine requirements now
8	Don't overstrain your hardware
9	Don't try to retrofit quality
10	Don't write your own test plans
11	Establish a software process that provides flexibility
12	Fix requirement specification errors now
13	Give product to customers early
14	Grow systems incrementally
3	Implement a disciplined approach and improve it continuously
16	Invest in understanding the problem
17	Involve the customer
18	Keep design under intellectual control
19	Maintain clear accountability for results
20	Produce software in a stepwise fashion
21	Quality is the top priority; long-term productivity is a natural consequence of high quality
22	Rotate (top performing) people through product assurance
23	Since change is inherent to software, plan for it and manage it
24	Since tradeoffs are inherent to software engineering, make them explicit and document them
25	Strive to have a peer find a defect, rather than a customer
26	Tailor cost estimation methods
27	To improve design, study previous solutions to similar problems
28	Use better and fewer people
29	Use documentation standards
30	Write programs for people first
31	Know software engineering techniques before using development tools
32	Select tests based on the likelihood that they will find faults
33	Choose a programming language to ensure maintainability
34	F aced with unstructured code, rethink the module and redesign it from scratch

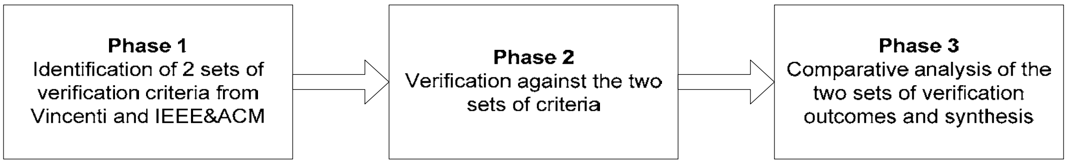


Figure 2. The three-phase verification process.

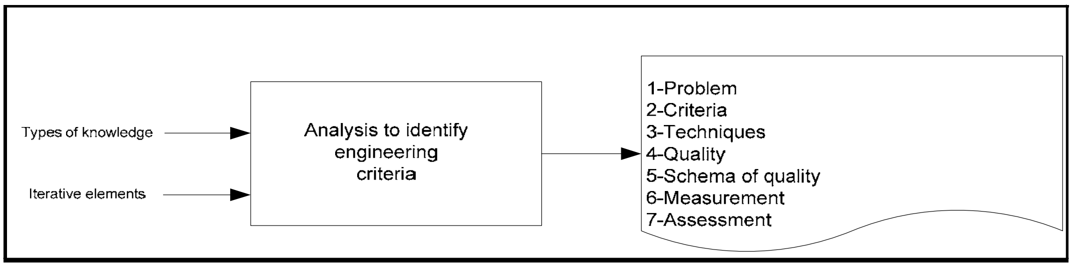


Figure 3. Identification of Vincenti engineering criteria.

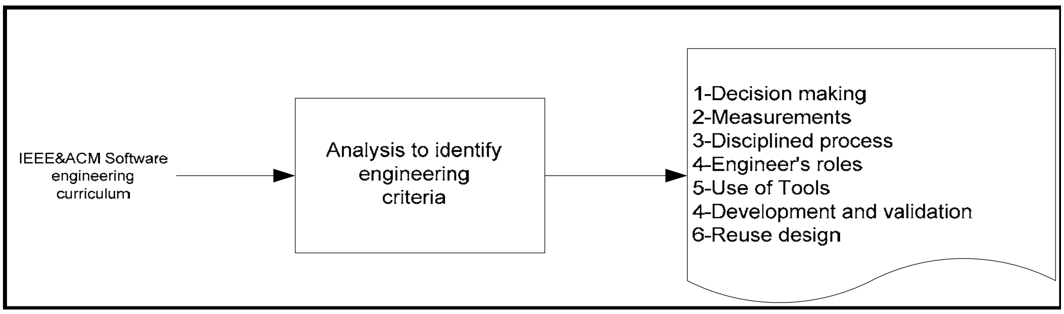


Figure 4. Identification of the IEEE-ACM engineering criteria.

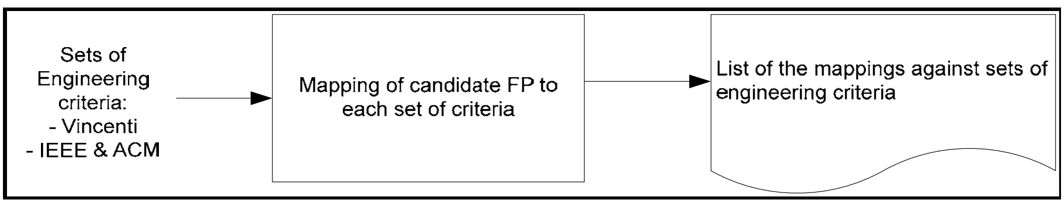


Figure 5. Phase 2: Process of verification against sets of engineering criteria.

6) Design instrumentalities.

According to Vincenti, this classification is not specific to aeronautical engineering, but can be transferred to other engineering domains. For instance, a detailed analysis of engineering knowledge types was used in [1] to analyze the content of the Software Quality Knowledge Area of the Guide to the Software Engineering Body of Knowledge–SWEBOK [13].

Vincenti has distinguished seven elements for engineering, which he refers to as "interactive elements", and which he selected prior to categories of engineering knowledge types. These elements show the epistemological structure of the engineering learning process based on the analysis of the five aeronautical case studies. These seven elements represent, in Vincenti's opinion, a necessary set of different elements that interact with each other for the completion of an engineering activity. These seven interactive elements are referred to here as the Vincenti engineering criteria, and are listed in **Table 2**. The abbreviations we have selected to represent each of these criteria are listed in the right-hand column of **Table 2**.

4.2. IEEE and ACM Joint Curriculum

The IEEE Computer Society (IEEE-CS) and the Association for Computing Machinery joined forces to develop a joint set of computer curricula, including one on software engineering. More specifically, chapter 2 of the joint software engineering curriculum lists the characteristics of an engineering discipline (see **Table 3**). These

Table 2. Engineering criteria identified in Vincenti.

ID.	Vincenti Engineering Criteria	Abbreviation
1	Recognition of a problem	Problem
2	Identification of concepts and criteria	Criteria
3	Development of instruments and techniques	Techniques
4	Growth and refinement of opinions regarding desirable qualities	Quality
5	Combination of partial results from 2, 3 and 4 into practical schema for research	Testing
6	Measurement of characteristics	Measurement
7	Assessment of results and data	Assessment

Table 3. Identification of IEEE & ACM engineering criteria.

ID.	Engineering Criteria Identified	Abbreviation
1	Engineers proceed by making a series of decisions, carefully evaluating options, and choosing an approach at each decision point that is appropriate for the current task in the current context. Appropriateness can be judged by tradeoff analysis, which balances costs against benefits.	Decision making
2	Engineers measure things, and, when appropriate, work quantitatively; they calibrate and validate their measurements; and they use approximations based on experience and empirical data.	Measurements
3	Engineers emphasize the use of a disciplined process when creating a design and can operate effectively as part of a team in doing so.	Disciplined process
4	Engineers can have multiple roles: research, development, design, production, testing, construction, operations, management, and others, such as sales, consulting, and teaching.	Engineer's roles
5	Engineers use tools to apply processes systematically. Therefore, the choice and use of appropriate tools is key to engineering.	Use of Tools
6	Engineers, via their professional societies, advance by the development and validation of principles, standards, and best practices.	Development and validation
7	Engineers reuse designs and design artifacts.	Reuse design

characteristics are adopted here as engineering verification criteria. The abbreviations we have selected to represent each of these criteria are listed in the right-hand column of **Table 3**.

5. Phase 2: Verification against the Two Sets of Criteria

The set of 34 candidate FPs is next mapped to the two sets of engineering criteria: each candidate FP is taken as input and analyzed using each of Vincenti's seven criteria and, again, each of the seven IEEE-ACM software engineering criteria.

The output of the mapping to Vincenti's engineering criteria is presented in Appendix A-1, where the letter D represents a direct mapping, and the letter I an indirect one. For instance:

- Candidate FP #2 (Apply and use quantitative measurements in decision making) maps directly to Vincenti's criterion #6 and indirectly to Vincenti's criterion #4.
- Candidate FP #31 (Know software engineering techniques before using development tools) has only an indirect mapping to criterion #3 and to criterion #7 (Assessment).
- Finally, there are candidate FPs with no mapping to any engineering criteria: for instance, candidate FP #13 (Give product to customers early).

This first verification against the Vincenti criteria leads to the following results (see Appendix A-1):

- 12 candidate FPs have at least one direct mapping to a Vincenti engineering criterion;
- 21 candidate FPs have only indirect mappings to Vincenti engineering criteria;
- 1 candidate FP has no direct or indirect mapping to any Vincenti engineering criteria.

The second verification against the seven IEEE & ACM engineering criteria is presented in Appendix A-2.

For instance:

- Candidate FP #2 (Apply and use quantitative measurements…) has a direct mapping to criteria #1 (Decision making) and #2 (Measurements). Candidate FP #16 (Invest in the understanding of the problem) is mapped indirectly to criteria #1 (Decision making) and #3 (Disciplined process).
- Candidate FP #4 (Build with and for reuse) is mapped directly and indirectly to criteria #7 (Reuse) and #3 (Disciplined process).
- Finally, candidate FP #13 (Give products to customers early) is not related to any engineering criteria.

This second verification against the IEEE and ACM criteria leads to the following results (see Appendix A-2):

- 15 candidate FPs have at least one direct mapping to an IEEE-ACM engineering criterion;
- 16 candidate FPs have only indirect mappings to an IEEE-ACM engineering criterion;
- 3 candidate FPs have neither direct nor indirect mappings to any IEEE-ACM engineering criteria.

6. Phase 3: Analysis and Consolidation Using Both Sets of Criteria

6.1. Analysis across Each Set of Engineering Criteria

The candidate FPs with a direct mapping to either the Vincenti or IEEE-ACM criteria are listed in **Table 4**. From a comparison of the two columns in this table, the candidate FPs with direct mappings can be grouped into three sets:

1) Candidate FPs with a Vincenti mapping similar to the IEEE-ACM mapping;

2) Candidate FPs with a Vincenti mapping with no equivalent IEEE-ACM mapping;

3) Candidate FPs with an IEEE-ACM mapping with no equivalent Vincenti mapping.

Table 4. Candidate FPs that directly meet criteria from either set.

#	Vincenti Mapping	#	IEEE-ACM Mapping
2	Apply and use quantitative measurements in decision making	2	Apply and use quantitative measurements in decision making
4	Build with and for reuse	4	Build with and for reuse
		5	Define software artifact rigorously
		6	Design for maintenance
7	Determine requirements now		
9	Don’t try to retrofit quality		
		10	Don’t write your own test plans
11	Establish a software process that provides flexibility	12	Fix requirements specification error now
14	Grow systems incrementally		
15	Implement a disciplined approach and improve it continuously	15	Implement a disciplined approach and improve it continuously
16	Invest in understanding the problem		
		18	Keep design under intellectual control
21	Quality is the top priority; long term productivity is a natural consequence of high quality	21	Quality is the top priority; long term productivity is a natural consequence of high quality
		22	Rotate (high performing) people through product assurance
23	Since change is inherent to software, plan for it and manage it		
24	Since tradeoffs are inherent to software engineering, make them explicit and document them	24	Since tradeoffs are inherent to software engineering, make them explicit and document them
		25	Strive to have a peer, find a defect rather than a customer
		26	Tailor cost estimation methods
27	To improve design, study previous solutions to similar problems	27	To improve design, study previous solutions to similar problems
		31	Know software engineering techniques before using development tools

Set A:

From **Table 4**, we can see that six candidate FPs (#2, #4, #15, #21, #24, #27) are present in both columns (the highlighted ones) and therefore satisfy at least one engineering criterion in each set of criteria (Vincenti and IEEE-ACM): these six could reasonably be considered as FPs that conform to engineering criteria.

Set B:

From **Table 4**, we note that there are 6 other candidate FPs that meet the Vincenti criteria, but no IEEE-ACM criteria, and 9 candidate FPs that meet the IEEE-ACM criteria, but no Vincenti criteria. Can these still be considered as FPs, or are they merely instances of more fundamental principles?

To answer this question, we could reasonably argue from the Vincenti subset that:

- Candidate FP #7 (Determine requirements now) can be deduced from candidate FP #16 (Invest in understanding the problem);
- Candidate FP #9 (Don’t try to retrofit quality) can be deduced from candidate FP #21 (Quality is the top priority; long-term productivity is a natural consequence of high quality);
- Candidate FP #11 (Establish a software process that provides flexibility) can be deduced from FP #9 (Don’t try to retrofit quality).

This would eliminate candidates FPs #7, #9, and #11 from the list of FPs, since they represent specific instantiations of more general FPs, while principles #16, #14, and #23 would be retained on the list of FPs.

Set C:

From **Table 4**, there remain 9 candidate FPs without a corresponding direct mapping to the Vincenti criteria. We could reasonably argue that these 9 can be deduced from those with direct Vincenti mappings: for instance, FP #18 (Keep design under intellectual control) and FP #31 (Know software engineering techniques before using development tools) can be deduced from FP #15 (Implement a disciplined approach and improve it continuously).

This would eliminate candidate FPs #5, #6, #10, #12, #18, #22, #25, #26, and #31 from the list of FPs, since they represent specific instantiations of more general FPs, while retaining principle #15.

In summary, this analysis has allowed us to refine the list of 34 candidate principles to 9 software engineering principles based on engineering criteria. This analysis has also eliminated the overlap between the various principles; as a consequence, a subset of only 9 (see **Table 5**) from the list of 34 candidates identified in Seguin 2006 are recognized as principles that conform to engineering criteria, the remaining 25 being specific instantiations of those 9. In **Table 5**, these software engineering FPs are sequenced from 1 to 9, along with their original sequence number (right-hand column) assigned when the initial list of 34 candidates was compiled.

6.2. Identification of a Hierarchy

Table 6 presents next the outcome of our analysis of the 25 remaining candidate FPs as instantiations of the 9 principles in **Table 5** that conform to engineering criteria.

7. Work Limitations

The initial list of 34 candidates taken as input for this research is not necessarily exhaustive: to summarize, these 34 candidates have been refined from 304 proposed principles identified in the literature over a period of 20 years, up to 2006 [15]. The methodology used engineering

Table 5. List of 9 software engineering principles.

#	Vincenti, IEEE-ACM mapping	
1	Apply and use quantitative measurements in decision making	2
2	Build with and for reuse	4
3	Grow systems incrementally	14
4	Implement a disciplined approach and improve it continuously	15
5	Invest in the understanding of the problem	16
6	Quality is the top priority; long term productivity is a natural consequence of high quality	21
7	Since change is inherent to software, plan for it and manage it	23
8	Since tradeoffs are inherent to software engineering, make them explicit and document it	24
9	To improve design, study previous solutions to similar problems	27

Table 6. Hierarchy of candidate principles for software engineering.

#	Direct mapping to Vincenti criteria		Derived instantiation (= Indirect mapping)
2	Apply and use quantitative measurements in decision making	26	Tailor cost estimation methods
		8	Don't overstrain your hardware
4	Build with and for reuse		
14	Grow systems incrementally	5	Define software artifacts rigorously
		20	Produce software in a stepwise fashion
15	Implement a disciplined approach and improve it continuously	1	Align incentives for developer and customer
		10	Don't write your own test plans
		17	Involve the customer
		18	Keep design under intellectual control
		20	Produce software in a stepwise fashion
		31	Know software engineering's techniques before using development tools
		19	Maintain clear accountability for results
		29	Use documentation standards
		10	Don't write your own test plans
16	Invest in understanding the problem	7	Determine requirements now
		12	Fix requirements specification error now
		17	Involve the customer
21	Quality is the top priority; long term productivity is a natural consequence of high quality	9	Don't try to retrofit quality
		22	Rotate (high performing people through product assurance
		25	Strive to have a peer find a defect rather than a customer,
		30	Write programs for people first
		3	Build software so that it needs a short user manual
		11	Establish a software process that provides flexibility
		28	Use better and fewer people
23	Since change is inherent to software, plan for it and manage it	6	Design for maintenance
		33	Choose a programming language to ensure maintainability
		32	Select tests based on the likelihood that they will find faults
		34	In the face of unstructured code, rethink the module and redesign it from scratch.
24	Since tradeoffs are inherent to software engineering, make them explicit and document them		
27	To improve design, study previous solutions to similar problems		

criteria from Vincenti and the joint IEEE & ACM software engineering curriculum to identify the 9 candidate principles that conform to these engineering criteria; however, this list of criteria is not necessarily exhaustive, and more criteria could eventually be added.

Furthermore, most of the authors who proposed these principles did not provide operational descriptions of their proposals, and did not provide research experimentation for each principle identified.

8. Conclusions

Software engineering, as a discipline, is certainly not yet as mature as other engineering disciplines, and, while a number of authors have proposed over 300 distinct FPs, a consensus on a set of well-recognized FPs has been lacking. This research report has taken as input, or as its object of study, the set of 34 statements identified in [15] as candidate FPs of software engineering. This set has been analyzed from an engineering perspective using the engineering criteria identified by either Vincenti or the IEEE-ACM joint effort on developing a software engineering curriculum.

The 34 candidate FPs were divided into three categories: A) candidate FPs that are directly linked to engineering, B) candidate FPs that are indirectly linked to engineering, and C) candidate FPs with no specific link to engineering.

In the next step, the candidate FPs that were generic were distinguished from the more specific ones: this distinction was based on the type of mapping (direct or indirect). In the final step, candidate FPs from both lists were analyzed and compared. Our proposed reduced list of 9 software engineering principles now needs to be further discussed by the software engineering community.

Of course, this list depends on the methodology used, and is being proposed to the engineering community for discussion and scrutiny with the aim of improving it and developing a consensus over time.

There is no claim that this list is exhaustive or that it covers the whole software engineering discipline. Even though the inputs to this analysis were derived from an extensive literature survey, this does not guarantee that those authors have indeed provided full coverage of the software engineering discipline.

Similarly, the hierarchy proposed in **Table 6** is derived from the engineering criteria in our analytical approach. Further research should be carried out to verify the completeness of the criteria used.

REFERENCES

[1] IEEE Std 610.12, IEEE Standard Glossary of Software Engineering Terminology. Corrected Edition, February 1991.

[2] K. E. Wiegers, “Creating a Software Engineering Culture,” Dorset House Publishing, New York, USA, 1996.

[3] N. Séguin, “Inventaire, Analyse et Consolidation des Principes Fondamentaux du Génie Logiciel,” *École de technologie supérieure*, Université du Québec, Québec, Canada, 2006.

[4] P. Bourque, R. Dupuis, A. Abran, J. W. Moore, L. Tripp and S. Wolff, “Fundamental Principles of Software Engineering–A Journey,” *Journal of Systems and Software*, Vol. 62, No. 1, 2002, pp. 59-70.

[5] Jabir, J. W. Moore, “A Search For Fundamental Principles of Software Engineering,” Report of a Workshop Conducted at the Forum on Software Engineering Standards Issues, Computer Standards & Interfaces–In- ternational Journal on the Development and Application of Standards for Computers, Data Communications and Interfaces, Elsevier, Amsterdam, North Holland (the participants at this workshop names their group “Jabir”), Vol. 19, No. 2, 1998, pp. 155-160.

[6] B. W Boehm, “Seven Basic Principles of Software Engineering,” *Journal of Systems and Software*, Vol. 3, No. 1, 1983, pp. 3-24.

[7] A. M. Davis, “201 Principles of Software Development,” McGraw-Hill, New York, USA, 1995.

[8] K. E. Wiegers, “Creating a Software Engineering Culture,” Dorset House Publishing, New York, 1996.

[9] P. Bourque and R. Dupuis, “Fundamental Principles of Software Engineering,” *Third International Symposium and Forum on Software Engineering Standards*, Walnut Creek, CA, USA, 1997.

[10] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, “Pattern Oriented Software Architecture,” John Wiley & Sons, England, 1996.

[11] C. Ghezzi, M. Jazayeri and D. Mandrioli, “Fundamentals of Software Engineering,” Prentice Hall, New Jersey, 2003.

[12] N. Séguin and A. Abran, “Inventaire des principes du génie logiciel,” *Revue Génie Logiciel*, Numéro 80, Paris, France, 2007, pp. 45-51.

[13] W. G. Vincenti, “What Engineers Know and How They Know It—Analytical Studies from Aeronautical History,” Johns Hopkins University, Baltimore, MD, USA, and London, UK, 1990.

[14] IEEE and ACM, “IEEE Computer Society and Association for Computing Machinery,” Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, A Volume of the Computing Curricula Series, 2004.

[15] SO-TR-19759, “Software Engineering—Guide to the Software Engineering Body of Knowledge (SWEBOK),” *International Organization for Standardization*, Switzerland, 2005.

[16] A. Abran and K. Meridji, “Analysis of Software Engineering from an Engineering Perspective,” *European*

Journal for the Informatics Professional, Vol. 7, No. 1, February 2006, pp. 46-52.

[17] A. Abran and J. W. Moore, P. Bourque and R. Dupuis, "Guide to the Software Engineering Body of Knowledge," 4th Edition, In: P. Bourque and R. Dupuis, Eds., *IEEE Computer Society*, Los Alamitos, CA, USA, 2005.

[18] A. Abran, N. Seguin, P. Bourque and R. Dupuis, "The Search for Software Engineering Principles: An Overview of Results," *Conference on the Principles of Software Engineering*, Buenos Aires, Argentina, 2004.

[19] P. Bourque and R. Dupuis, "Fundamental Principles of Software Engineering," 3*rd International Symposium and Forum on Software Engineering Standards*, Walnut Creek, CA, 1997.

J. Software Engineering & Applications, **2010, 3, 983-989**
doi:10.4236/jsea.2010.310115 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Towards an Efficient Information Systems Development Process and Management: A Review of Challenges and Proposed Strategies

Achimugu Philip[1], Babajide Afolabi[2], Oluwaranti Adeniran[2], Oluwagbemi Oluwatolani[1], Gambo Ishaya[2]

[1]Computer Science Department, Lead City University, Ibadan; [2]Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife.
Email: {check4philo, tolapeace, igpeni}@yahoo.com, {bafox, aranti}@oauife.edu.ng

Received July 31st, 2010; revised August 26th, 2010; accepted August 31st, 2010.

ABSTRACT

Before Information Systems are developed, they must have undergone a process called Systems Development Life Cycle (SDLC) using appropriate methodology. The SDLC consists of phases varying from author to author. However, an information systems project can only be successful with intense interaction amongst project manager, systems analyst, system designers and the end users. Viewed from the project manager's perspective, the SDLC lacks the essence of project management activities. Similarly, end users involvement is not clearly specified. The main aim of this paper is to propose a framework for information systems management and development process which accommodates the views of the different participants. Furthermore, the paper sharpens the concept of conventional SDLC, on the basis of the proposed framework. In addition, tools and methods that are appropriate for the implementation of the framework are herein discussed.

Keywords**:** *SDLC, Information Systems, Framework, Project Management, Development, End Users*

1. Introduction

The early applications of computers were implemented without the aid of any explicit Information Systems (IS) development methodology and appropriate management techniques. In these early days, the emphasis of computer applications was towards programming. This meant that system developers were technically trained but were not necessarily good communicators. This often meant that the needs of the users in the application area were not well established, with the consequence that the IS design was frequently inappropriate for the application. Few programmers would follow any formal methodology; in most cases, they use rule-of-thumb and rely on experience [1].

Estimating the date on which the system will be operational was difficult and applications were frequently behind schedule. Programmers might spend a very large proportion of their time on correcting and enhancing the applications which were operational. Typically, a user will come to the programmer asking for a new report or modification of one that was already supplied. Often, these changes had undesirable effects on other parts of the system, which also had to be corrected. This vicious circle will continue, causing frustration to both programmers and users. As computers increased rapidly in number and management was demanding more appropriate systems for their expensive outlay, the situation could not continue. There were three main changes [2]:

1) The first was a growing appreciation of the part of the development of the system that concerns analysis and design and therefore, the role of the system analyst as well as that of the programmer;

2) The second was realizations that as organizations were growing in size and complexity; it was desirable to move away from one-off solutions towards a more integrated approach;

3) The third was an appreciation of the desirability of an accepted methodology for the development of information systems.

Organizations today are much more concerned about the effects of competition than they were in the past;

therefore, no organization would like to stand the risk of being overtaken by other competitors on the same playing ground with equal opportunities. Organizations that acquire prompt delivery of information system projects and posses efficient management skills will always be at the fore front of this global digital drive which commands profits for organizations and good quality of services for users and customers. Although, traditional uses of information technology still exist, new information systems development has become one of the most important weapons for organizations to gain competitive advantage. New application development is the most vigorous for those organizations that recognize information as a resource for achieving their strategic goals.

Existing literature provides some formal methods and management models for information systems development which cannot explain all the tasks that must be performed by the diverse group of people that are involved in the development process of information systems. For instance, the waterfall model in isolation cannot fully explain the perspective of the project manager, same goes to the capability maturity model and hosts of others. The primary management goal is to build a working information system under a planned budget and schedule. The activities such as planning, organizing, staffing, leading and controlling are of particular importance in managerial activities [3].

The main aim of this paper is to propose a framework for an efficient information systems development process and management that will enable information system projects to be promptly and successfully completed through the integrated efforts and view of the project manager and end users, along with other project staff members such as system analysts, developers, programmers and maintenance programmers.

2. Information Systems Management and Development Process

New information system development typically starts with a temporary organizational structure called project team. Typically, a project team consists of a project manager, system analyst, programmers, etc. A project manager, usually a senior system analyst in the organization has the responsibility of the entire project. The project members must intensively interact with users. For prototyping projects, the team must include the users. The importance of users' participation in information systems development is highlighted by an increasing use of new software productivity tools such as Computer Aided Software Engineering (CASE) tools. These tools enable users to be actively involved in the system development process, and to improve the chance that the final system will be adopted by users. It therefore must be emphasized that information systems can be successfully completed only with intense interactions among project participants. A critical analysis of information system management and development process suggest that its different aspects should be highlighted according to different participants. Therefore, the management and development process is divided into three levels, each of which corresponds to a type of participant. A set of activities that should be performed at each level is defined as a schema. The hierarchical architecture consists of thee schemas: manager's, actor's and user's schemas.

2.1. Manager's Schema

This schema represents a set of activities performed by a project manager. Proper project management is a necessary ingredient for successful project implementation. The project manager must effectively use the management tools for proper project management. The project management goal must be achieved through appropriate management activities. These managerial activities are categorized into five functions: planning, organizing, staffing, leading and controlling. Each activity of a manager can be readily placed within one of these five management functions. This perspective is what a project manager must conceive during the course of project management. This level corresponds to manager's schema. The logical view of information systems conceived by each individual actor must be mapped into the manager's schema. In other words, each activity performed by actors must confirm to the manager's goal. This mapping also can ameliorate communication barriers among project developers and manager.

2.2. Actor's Schema

This schema represents the activities assigned to system analyst, programmers and maintenance programmers. These developers, except for managers will be referred to as actors. The primary goal of actors must be to meet user's requirement as spelt out in the Software Requirement Specification (SRS) document. The actor's goal is to develop the information system that will be successfully adopted by end users. This schema explicitly encompasses the SDLC, from system planning to maintenance. A set of activities at this level is referred to as actor's schema.

2.3. User's Schema

This schema represents the activities performed by end users; it must aid actors to develop a successful system.

The user's schema represents a set of activities by end users. The main purpose of this schema is to identify and describe the tasks that are required of users to guarantee the success of the project. Users and actors often have a many-to-many relationship. In other words, one actor may work with several users and one user may work with several actors. The user's schema must be mapped into the requirements.

The hierarchy is useful for describing a concept for information systems management and development. Any information system project can neatly be matched into the prototype framework. The primary advantages of the three schema hierarchy are highlighted as follows:

First, the information systems development process can be better understood by project participants. Better understanding of other participant's perspective of the development process results in improved communication among project members along with users. The communication gap has been the most significant cause of system failures. Secondly, the hierarchy views the entire development process as an integrated entity. It integrates different views and thus reduces the task duplication. Furthermore, the hierarchy sharpens the idea of how the information requirements can be mapped into strategic goal for the information systems in an organization. It also presents mapping between users and actors. This mapping is significant especially for prototyping approach.

The three schema hierarchy can accommodate both the top-down view and the bottom-up view of the systems management and development activities. Obviously, systems management and development consist of both management and development processes. Typically, the management process in isolation starts with an activity by a manger. The activities performed by a manager must be mapped into activities with actors and then activities by users. This transitive specialization corresponds to a top-down view for systems development. In contrast, to management process, the development process starts with the activities performed by the users. The activity performed by the users must be triggered by users' requirements. The activities are mapped into activities by actors and then activities by managers. This specialization corresponds to a bottom-up view for system development. The hierarchy also highlights the user participation in the project. It is of particular importance for prototyping approaches. The role of SDLC in prototyping is sharpened by a specific description of user's involvement. This improvement is of particular interest because the traditional SDLC discourages more effective approaches like prototyping.

The aim of presenting a three schema hierarchy is to provide a framework in which each individual project participant can better conceive the overall view of the information systems project. Furthermore, each project participant can better perform his task so that the system can be finally accepted within the organization.

3. Conceptual Framework for Enhanced Information Systems Development and Management Process

From a systems analyst's perspective, the SDLC perfectly illustrates the systems development process. Typically, the SDLC consist of several phases. Information systems literature has produced a variety of SDLC phase's nicknames. The actual name of the SDLC phases may vary depending on authors. They generally differ in how many phases are recognized. However, the current SDLC concept displays some shortcomings. For instance, the life cycle concept aggravates the communication gap between end users and actors. It also fails to specify the interdependency between a manager and the actors. In order to address these problems, we have incorporated the system management process into the enhance ISD and management framework we have proposed in this report. The framework is presented below:

The outset situation of the IS investment project consists of the organizational norms and values, project specific contextual factors and the resources given to the project. The actual outcomes of the IS investment project are produced in conjunction with the business development process, the IS development and procurement processes. The outcomes of the IS project are defined by the success of the system with respect to the investment perspective, the success of the IS project implementation and the success of the desired IS functionality. The basic idea in our conceptual framework, presented in **Figure 1**, is the inclusion and integration of all IS participants involved in its development and management processes throughout the IS product life cycle. The components of the framework are discussed in more details in the following.

3.1. Outset Situation

Organizations operate and survive through organizationally accepted rules that are justified by goals or a hierarchical goal system. Within organizations, there are individual goals, objectives, desires, wishes, intentions, etc, as well as organizational goals, objectives, missions, etc. [4]. As pointed out by [5], any strategic investment process employs individual and organizational values and preferences, goals and objectives as an input. At best, undertaking this task will help the project team members to understand the organization's processes, problems and

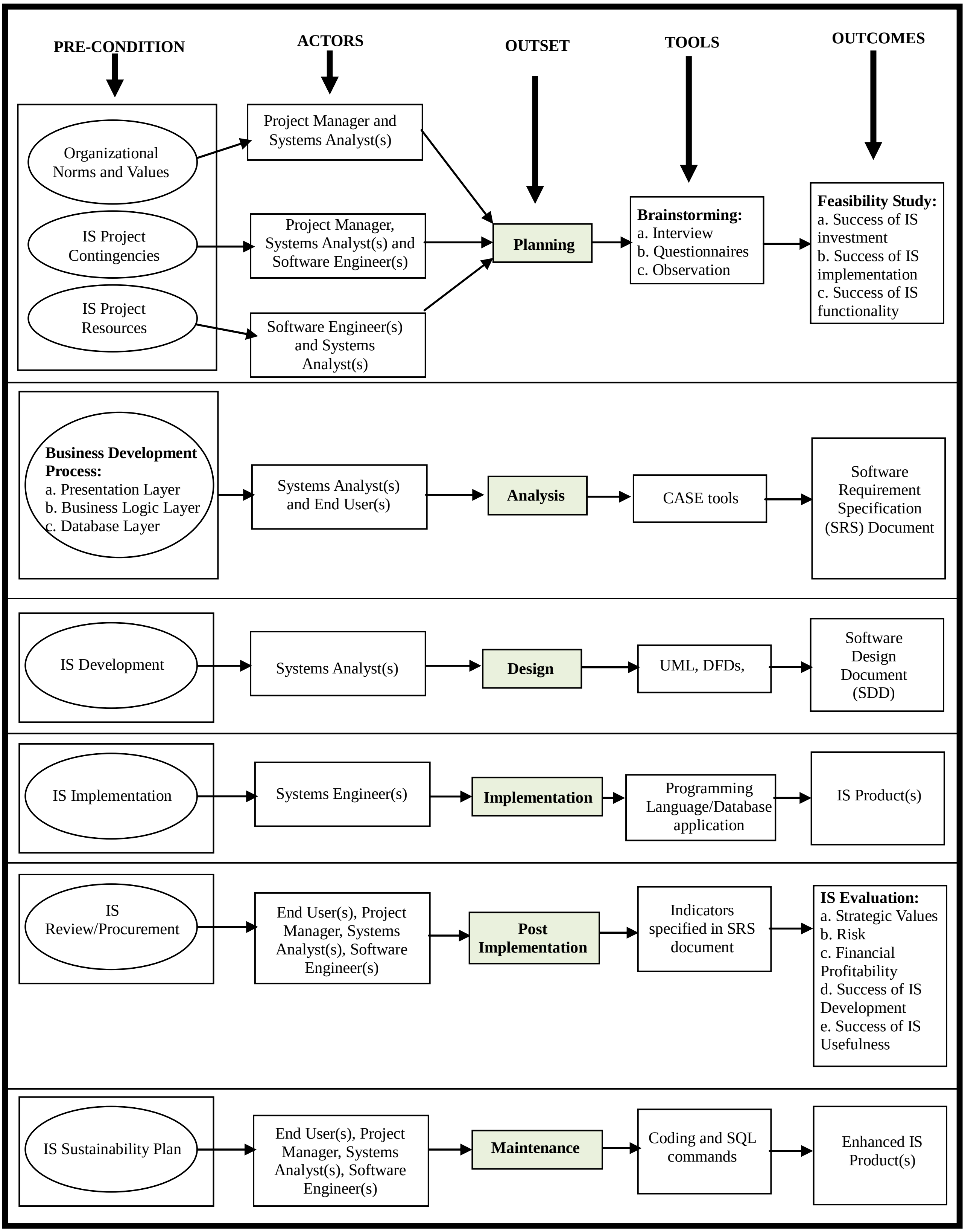


Figure 1. Conceptual framework for enhanced information systems development and management process.

opportunities, thus facilitating organizational learning.

3.2. Project Contingencies

An information system can, on the one hand, be a small application supporting only one single activity, but on the other hand, it can be a wider system supporting the whole company, or it can even be an inter-organizational system. There is one additional type of information system that deserves special attention, which are infrastructure investments. Infrastructure investments are of high importance because they create the platform on which future applications can be built. Moreover, why the information system is actually built, depends on several factors. In some situations a company may be forced to build a new information system, e.g. because of legislation changes. Additionally, the senior management may perceive that the system needs to be built, for example, to support a business strategy. Finally, the arguments for building the system can be from the expected and clear quantitative or qualitative benefits from the investment perspective.

The nature of the investment varies according to the novelty of the system. An investment can deal with improving an existing system, replacing an old system or developing an entirely new system. Reference [6] asserts that the nature of the investment differs according to how common this type of system is in the field of industry where the company operates. For example, investment in a routine system is different from an investment in an innovative system. All these must be taken into cognizance for an effective and efficient IS development and management process at the planning stage of the life cycle.

3.3. IS Project Resources

Both the material and the immaterial resources are crucial while developing information systems. The integrating role of the major actors as depicted in figure 1 includes detecting possible problems and as a result of evaluation it may be noticed, for example, that the project needs more system development resources.

3.4. Business Development Process

As information technology can make alternative operational designs possible, it in many cases plays a central role when developing the company's business strategy that is to be embedded in the IS product in order to improve organizational processes. Furthermore, IT enables new kinds of flexible inter-organizational arrangements. Moreover, information technology can support the development of new business, or new products and services. Thus, IS projects are often connected to larger strategic business development programs and the role of IS actors under this category is to ensure that the IS project would deliver the required technological capabilities for achieving the strategic business objectives. The degree that an IS project is involved in business development can range from a system that supports the current business strategy to a system that creates competitive advantages and new business opportunities to the company. An information system investment is in many cases an important part of a business process re-engineering project. The actual aim of the system investment would be cutting costs, improving products or services, or serving a certain customer group better.

3.5. IS Development Process

In the ever-changing business environment, it is very important to be aware of the possible changes that may, in some situations, affect the underlying assumptions that the investment is based on. Thus, it would be essential to conduct evaluation regularly during the development process. According to [3], there are basically three development strategies: to use a system life cycle -based methodology, an iterative methodology (e.g. prototyping) or a mixed methodology. The choice of the development method obviously affects the way evaluation is conducted; for example, prototyping can be considered an evaluation methodology in itself. There are several factors that affect the risk of the development process. First, the risks would be decreased if parts of existing systems or existing knowledge can be exploited in system development [3]. Secondly, there are some factors related to project management; the knowledge and skills of the system developers and the representatives of the users affect the risks of the IS development project; the cooperation within the project group and between the project group and the users must be active in order to minimize the project risks; and the risks of the project could be decreased by using formal project management and control methods [3].

In summary, reference [1] presented the following factors affecting the risk of a system development project: 1. Technological newness; 2. Application size; 3. Expertise of development team and users; 4. Application complexity; 5. Organizational environment (e.g. conflicts, role definitions).

3.6. IS Procurement Process

Basically, an information system may be developed in-house (custom written), it may be developed by a software vendor, or the company may purchase a software package (commercial off-the-shelf). Reference [7] described two recent trends in information resource acquisition: firstly, the process has changed from an internal to market-oriented; second, there is a more distinct focus on business processes. When using an outside vendor to develop the system, evaluation procedures should be explicitly designed for contracting purposes, since all individuals acting as clients for IS projects may not be knowledgeable about the technology related issues. While IS often plays a central role in developing new business processes, the choice of the IS procurement strategy is critical for company operations. For different kinds of systems different kinds of resources are needed and consequently different procurement strategies are applicable. According to the procurement principles for choosing the efficient procurement strategy, presented by [6], routine systems should be implemented by acquiring software packages from implementers, while standard applications require software contracting by analysts and possibly other outside resources for implementation, and speculative investments are best left for internal development by innovators.

3.7. IS Evaluation Process

Reference [8] argued that, the evaluation process should identify and control the critical areas of an IS project. Before selecting the evaluation criteria and methods and deciding who would be involved in the evaluation, it is important to identify all the relevant interest groups for the IS project. A covering set of evaluation criteria should be used to make sure that all dimensions of the IS endeavour are taken into account and assessed. The IS evaluation process must be integrated into business development process, the IS development process, and the IS procurement process.

Reference [9] suggested a three-step process for IS evaluation: 1. Intangible benefits evaluation, 2. IS investment risk analysis, and 3. Tangible benefits evaluation. The steps should be taken in this order, i.e., intangible benefits and risks should be evaluated prior to evaluating the tangible benefits. In our framework, the order of the evaluation categories “strategic value”, “risks” and “financial profitability” reflects this suggestion. The “success of IS development” category is placed prior to the “success of IS usefulness” since the usefulness can only be observed after the IS has been used for a while. Ideally, IS evaluation would cover all the above categories, but, however, it is expected that the focus of evaluation is different depending on who conducts the evaluation and where the initiative for the evaluation comes from. Reference [5] stated that the focus of evaluation changes according to the organizational interests, which may be on a number of levels, e.g. costs and benefits, organization’s competitive position or industrial relations. We argue, however, that whether the organization’s interests are taken into account appropriately depends on the knowledge and skills of the evaluator. Thus, the senior management should carefully consider who should be involved in the evaluation. The result of the evaluation should be delivered to each person related to the project so that the information received from the evaluation can be employed in the decision making phase. Most likely, the decision itself would be continuing with the investment (maybe after some minor changes), changing the specifications, range or implementation method of the system, or ‘freezing’ the project. In addition, the changes might include e.g. schedule changes; reorganization of the project (e.g. project management can be changed); or vendor changes. The reasons for these changes may be obvious mistakes, unexpected problems, a new experience about the project that changes the idea of the right course of action, or changes in the company's environment, that are beyond the company’s control.

3.8. Outcomes

The outcomes of an IS project are identified as the success of 1) IS implementation, 2) IS investment, and 3) IS functionality. IS Evaluation should not work only as a justification mechanism but as a tool for experience learning. During the IS development process, feedback from the evaluation process should lead to corrective actions if necessary. These actions might include, for example, a change in the information system development or procurement strategy, or a change in the resources that are given to the project.

Saarinen [10] noted that evaluating the success of an IS implementation should consider at least two dimensions: the process and the product success. Evaluating the conduct of the IS development process would facilitate the learning for future projects. The product success includes both the IS functionality and the realization of the expected benefits from the IS investment. Hallikainen Heikkilä, Peffers Saarinen, and Wijnhoven, reference [11] puts it that to learn conducting evaluation and managing information system projects more effectively, the perceived success of the evaluation process itself can be measured in terms of: evaluation efficiency, precision, and effectiveness. Evaluation efficiency can be divided into efficiency of evaluation process and cost of evaluation. Evaluation precision can be further divided into satisfaction with evaluation criteria and methods used; and satisfaction with contents, usability and reliability of information produced by evaluation. Finally, evaluation effectiveness can be divided into usefulness of the results of evaluation when making decisions concerning this particular project; and evaluation supporting in aligning information technology and business functions.

4. Conclusions

Our proposed framework is useful in describing a concept for information systems management and development. Any information system project can neatly be matched into our proposed framework. The primary advantages of our framework are highlighted as follows:

First, the information development process can be better understood by project participants. Better understanding of other participant's perspective of the development process results in improved communications amongst project members along with users. The communication gap has been most significant cause of system failures. Second, our framework views the entire development process as an integrated entity. It integrates different views and thus reduces the task of duplication. Thirdly, our proposed framework sharpens the idea of how the information requirements can be mapped into the strategic goal for the information systems in an organization. It also provides a mapping between users and actors (developers). This mapping is significant especially for prototyping approach.

Finally, our framework accommodates both top-down view and the bottom-up view of the systems management and development activities. Typically, the management process in isolation starts with an activity with by a manager. Activities performed by the manager must be mapped into activities by the actors followed by activities by the users. This transitive specialization corresponds to top-down view for systems development. On the contrary, the development process starts with the activities performed by the users which must be triggered by the users' requirements. These activities are mapped into activities by actors and finally activities by manger. This specialization corresponds to a bottom-up view for systems development. The conventional SDLC phases is however, sharpened by the incorporation of clear cut roles assigned to users during the life cycle of an IS product.

REFERENCES

[1] H. Barki, S. Rivard and J. Talbot, "Toward an Assessment of Software Development Risk," *Journal of Management Information Systems*, Vol. 10, No. 2, 1993, pp. 203-225.

[2] P. Barrow and P. Mayhew, "Investigating Principles of Stakeholder Evaluation in a Modern IS Development Approach," *The Journal of Systems and Software*, Vol. 52, No. 2-3, 2000, pp. 95-103.

[3] G. B. Davis, "Strategies for Information Requirements Determination, IBM," *Systems Journal*, Vol. 21, No. 1, 1982, pp. 4-31.

[4] H. Kivijärvi, "Knowledge Conversion in Organizational Context: A Framework and Experiments," *Proceedings of the 37th Hawaii International Conference on System Sciences*, USA, 2004, p. 10.

[5] D. Santos and B. L. David, "Justifying Investments in New Technologies," *Journal of Management Information Systems*, Vol. 7, No. 4, 1991, pp. 71-90.

[6] P. Hallikainen, H. Kivijärvi and K. Nurmimäki, "Evaluating Strategic IT Investments: An Assessment of Investment Alternatives for a Web Content Management System," *Proceedings of the HICSS-35*, *Big Island*, Hawaii, 2002, pp. 2977-2986.

[7] T. Saarinen and A. Vepsalainen, "Procurement Strategies for Information Systems," *Journal of Management Information Systems*, Vol. 11, No. 2, 1994, pp. 187-208.

[8] R. Heckman, "Managing the IT Procurement Process Information," *Systems Management*, Vol. 16, No. 1, 1999, pp. 61-71.

[9] V. Serafeimidis and S. Smithson, "Information Systems Evaluation in Practice: A Case Study of Organizational Change," *Journal of Information Technology*, Vol. 15, No. 2, 2000, pp. 93-105.

[10] H. J. Wen and C. Sylla, "A Road Map for the Evaluation of Information Technology Investment," In: Mahmood and Szewczak, Eds., *Measuring Information Technology Investment Payoff*: *Contemporary Approaches*, Idea Group Publishing, Addison Wesldy, New York, 1999, pp. 142-150.

[11] T. Saarinen, "Success of Information Systems. Evaluation of Development Projects and the Choice of Procurement and Implementation Strategies," PhD Dissertation Helsinki School of Economics and Business Administration A-88, 1993.

[12] P. Hallikainen, J. Heikkilä, K. Peffers, T. Saarinen and F. Wijnhoven, "Evaluating Information Technology Projects in Finland: Procedures, Follow-through, Decision-Making and Perceived Evaluation Quality," *Journal of Global Information Management*, Vol. 6, No. 4, 1998, pp. 23-33.

J. Software Engineering & Applications, **2010, 3, 990-997**
doi:10.4236/jsea.2010.310116 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

Design and Development of Virtual Objects to be Used with Haptic Device for Motor Rehabilitation

Jatin B. Maniya, Pushparaj Mani Pathak, Bhanu Kumar Mishra

Mechanical and Industrial Engineering Department, Indian Institute of Technology, Roorkee, India.
Email: pushpfme@iitr.ernet.in

Received August 2nd, 2010; revised August 25th, 2010; accepted September 2nd, 2010.

ABSTRACT

This paper presents design and development of virtual objects to be used with haptic device PHANTOM Omni for motor rehabilitation by incorporating visual and haptic feedback. Developed predominantly for augmentation of motor skill of patient, the objects could also be used for teaching and enhancing writing skill of children. The virtual objects include English alphabets, numbers, string and mazes, and games. The scheme has been evaluated on children from 11 *to* 14 *years age. The experiment gives good results.*

Keywords**:** *Haptic, Motor Rehabilitation, Virtual Environment*

1. Introduction

Haptic is derived from the Greek verb "haptesthai" meaning "to touch" and it refers to the science of touch and force feedback in human-computer interaction. As haptic devices are evolving and getting cheaper and more compact in size options are explored by the researchers for their use in medical applications. Haptic device provides an excellent addition to virtual environment due to the force feedback and tactile input/output when it is used for rehabilitation of patients suffering with motor disorder. This property also helps the children who are learning to write, who otherwise get this information from teacher's hand when he/she holds their hand.

Generally, stroke sufferer loses the skill of writing and they are unable to write again. From literature it is observed that little work has been carried out on enhancing writing skill of patient recovering after stroke. Many stroke victim state loss of handwriting ability as one of their biggest disabilities compared to other injury [1]. In 1998, Prisco, Avizzano, Calcara, Ciancio, Pinna and Bergamasco [2] have presented an immersive Virtual Environment (VE) including visual, auditory and haptic feedbacks, which had been designed specifically to help in recovery or improve the motor dexterity of the arm and hand of patients affected by motor disorder. They created kitchen base environment in which subject has to execute task like burning stove, putting coffee pot on stove etc. and they got satisfactory result when tried on patient. During the same time Henmi and Yoshikawa [3] developed virtual calligraphy system to transfer skill of teacher to student. They concluded that the student's position and force trajectories resemble closer to those of the teacher's as the exercise advances. Gillespie, O'Modhrain, Tang, Zaretzky and Pham [4] developed virtual environment, which can demonstrate certain object manipulation techniques. Although average performance time for the experimental groups did not improve faster than that of the control group, the optimal strategy was successfully communicated to the experimental groups. Pernalete, Yu, Dubey and Moreno [5] has developed an assessment and training procedure that result in improving handwriting taking advantage of the force feedback provided by the haptic device, incorporating inertia and viscosity effects to decrease the tremor in the hand as well as to stimulate the muscles. Teo, Burdet and Lim [6] introduced a virtual teaching system for Chinese ideograms that guides movement by haptic and visual feedback. It resulted in improvement of performance, in particular for beginners learning Chinese writing with a pen. Learning was evaluated by considering accuracy and speed of writing. Feygin, Keehner and Tendick [7] investigated use of haptics for skills training. Their study indicated that haptic guidance is effective in training. The performance was improved in terms of position, shape, timing, and drift. In 2005, Pemalete *et al.* [8] performed a test on a force-reflecting haptic interface device, PHANTOM with the GHOST SDK Software.

They used these devices to improve the hand eye coordination problem of children. In 2005, Mullins, Mawson and Nahavandi [9] have developed a method of controlling a user's hand through the process of writing. They introduced Proportional Differential (PD) controller that decreases the overshoot and reduces settling time to acceptable boundaries in PHANTOM. Grantner, Gottipati, Pernalete, Fodor and Edwards [10] have developed an intelligent decision support system for eye-hand coordination assessment which is based on fuzzy logic algorithm. It takes decision on whether subject should be allowed to move to next stage or not based on his/her previous performance. In 2007, Mansour, Eid, and Saddik [11] presented a multimedia system designed to facilitate learning of alphabetical handwriting of various languages by incorporating visual, auditory, and haptic feedback. Gouy-Pailler, Zijp-Rouzier, Vidal and Chêne [12] have represented a system consisted of a haptic pen and computer generated virtual environment. This system was used to teach the children, especially visually impaired children. This system was specially designed to teach geometric lessons. Subjects were asked to regenerate the shapes immediately after training. Experiments performed on them have shown impressive results, specifically in case of totally blind subjects. Palluel-Germain *et al.* [13] have developed a system incorporating a visuo-haptic device 'Telemaque' to increase the fluency of handwriting production of cursive letters in kindergarten children *i.e.*, before formal handwriting learning begins. Rassmus-Grohn, Magnusson and Efiring [14] presented the evaluation of an audio-haptic editor and explorer for virtual 2D relief drawings that allow visually impaired users to create and explore graphical images. In 2007, Kayyali, Shirmohammadi, Saddik and Lemaire [15] presented a system that uses haptics, in conjunction with virtual environments, to provide a rich media environment for motor rehabilitation of stroke patients. The system also provides occupational therapists with a human-computer interface that allows for easier set-up, more facilitated interaction with the system, and provides a more autonomous means for the progression of the patient. The system had been modified, after thorough analysis by a group of experienced occupational therapists. In [16], Bayart, Pocheville and Kheddar have presented a progressive guiding system designed as a software module for the framework I-TOUCH using simple XML code. They have built features like fully, partially and no guidance from haptic device, PHANTOM Omni. Then from the experiments carried out over three groups, they concluded that the one group given partial haptic help, perform well compared to other two.

This work presents virtual environment where objects, numbers, words and games have been created in open platform. The earlier work reported has been related with Chinese character [6]. Here we have created virtual objects that comprises of complete English alphabets, English numerals from 0 to 9, English words, and games. These virtual objects are interfaced with SensAble PHANTOM Omni haptic device. Presented system can provide training to enhance writing skill of motor stroke patients and children. This method of haptic training has the potential to stimulate re-learning in the brain of stroke victim. Alphabets, numbers and full words have been created with aim of increasing the learning ability whereas maze has been created to improve coordination skill in subjects. A ball tracking game has been created with aim of increasing the concentration of subjects. These objects have been created based on a study carried out on motor disordered patients. To validate the idea experiments have been carried out on subjects aged 11-14 years. The experimental results clearly indicate improvement in the learning ability of the subjects. The principal contribution of this work is logical creation of virtual objects which have been linked with PHANTOM Omni haptic device and testing of these objects on subjects for evaluating learning level.

2. Brief Description of Haptic PHANTOM Omni

The PHANTOM Omni model is the most cost-effective haptic device available today. Three degrees of force, in the *x*, *y* and *z*, direction are achieved through motors that apply torque at each joint in the robotic arm. The position and orientation of the pen are tracked through encoders in the robotic arm. **Figure 1** shows setup for rehabilitation.

Figure 1. Haptic Rehabilitation Setup with PHANTOM Omni and Computer.

3. Result of Study on Motor Disordered Subjects

Some subjects were studied to find major difficulties faced by them. This information was used to create virtual environment that can be used for augmentation of their motor skills. **Table 1** shows the details of the subjects, difficulties faced by them and the reason for the difficulty. Conclusion from **Table 1** is that motor disordered patients suffer from uncontrolled hand movement, due to which they cannot write properly.

4. Design of Virtual Objects for Training

The system consists of alphabets, numbers, string, maze and game. Presented system has been designed to give partial guidance to the subject. This system has different exercises on alphabets and numbers which are partially guided. Here partially guided means object will not react until subject is moving his hand within range of pre-specified path. When it tries to move outside of object, system will exert force on stylus and will keep it in range only. Thus, the system is neither fully guided nor fully free. To attain this feature in letters, we created 3-D model of letters which consists of walls.

In present system, Open Graphic Library Utility Toolkit (GLUT) is used for graphical presentation on screen. Haptic Device Application Programming Interface (HDAPI) provides low-level access to the haptic device. It also enables haptics programmers to render forces directly, offer control over configuration, the runtime behavior of the drivers. Haptic Library Application Programming Interface (HLAPI) provides high-level haptic rendering and allows significant reuse of existing OpenGL code and greatly simplifies synchronization of the haptics and graphics threads.

Table 1. Difficulties faced by disordered subjects.

S.N.	Subjects	Type of difficulty	Reasons
1	Subject A; Age-14 Years	Judgment about space and time, poor handwriting, language impairment	Motor disorder from childhood
2	Subject B; Age-19 Years	Uncontrolled body movement, could not stand still	Motor disorder developed after teenage
3	Subject C; Age-53 Years	Could not perform normal domestic work, could not write	Stroke victim
4	Subject D; Age-35 Years	Drop things frequently, awkward gait pattern	Motor disorder developed after teenage
5	Subject E; Age-70 Years	Problem with fine motor skill like cutting food, getting cup of tea, frequent fatigue	Faced motor disorder after 65 years

4.1. Alphabet and Number System

With the help of alphabet and number system virtual object, one can learn to write alphabet and numbers. The letters has been created in PRO/ENGINEER (Pro/E), solid modeling software. Using this model along with OpenHaptic Toolkit, we created scenario in which one can feel the shape of particular letter and move the cursor along with it. As the subject moves his hand through the shape, the stylus will not exert any reaction force on subject hand. But when stylus tries to run out of the wall of letter, subject will feel reaction force and it could not move beyond that limit. This ensures that subject hand remains within certain limit and thus the subject motion is neither free nor fully constrained. We also aided visual feedback through picture, which helps subject to remember used letter and correlate that picture to specific word. The flowchart used for creation of a letter is shown in **Figure 2**. First of all letters or numbers are modeled in Pro/E environment. The created file is in .prt format. This created file is converted in to .3ds format. Then model is imported in visual C++ environment. The properties of object like mass, stiffness, damping, friction etc. are set. Add picture of the object as visual aid. Also spelling of the object is added. The created alphabet *Q* is shown in **Figure 3**.

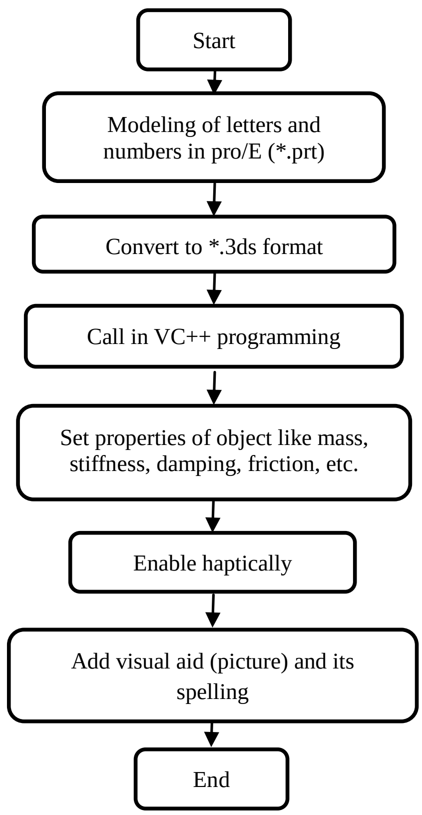


Figure 2. Flowchart for alphabet.

Creation of numbers is similar to alphabets. Its window contains a number and its spelling printed on a sphere. The snapshot of developed object is shown in **Figure 4**. The algorithm followed for its construction is similar to that used for alphabets.

4.2. String

After one gets practice on letters and number system, the next stage is to learn group of letters *i.e.* words. So, next set of exercise can be practice on strings. String is constructed using constrained motion scenario. The flowchart used for creation of string is shown in **Figure 5**. The screen shot of created string is shown in **Figure 6**.

4.3. Constrained Wall Maze

One can practice on constrained maze exercises and can build ability to take decision and follow the path. The created virtual object will not react if subject moves his hand on pre-defined path, but when he/she tries to deviate from it, he/she feel that the motion in that direction is constrained. Thus in this exercise subject could not move his hand in any direction but only pre-specified way. To

Figure 3. Alphabet exercise.

Figure 4. Letter exercise.

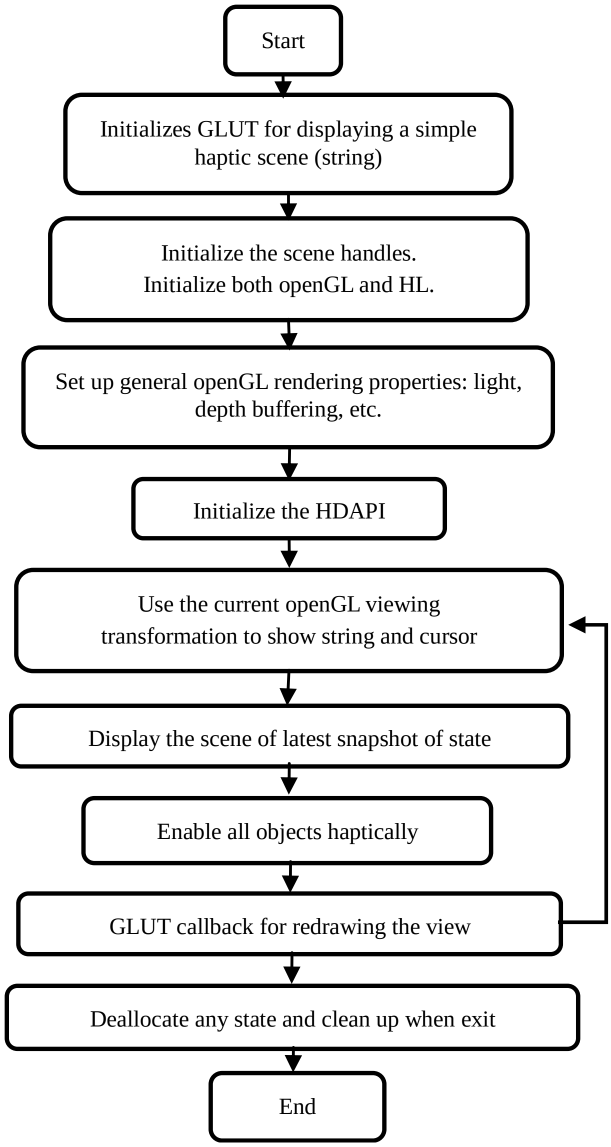


Figure 5. Flowchart for string.

Figure 6. Word (string) exercise.

reach at finish point from start point, subject has to take many decisions about the path which he has to choose to reach at end point. This exercise will help to augment decision making skill of subject. The flow chart used for creation of maze is similar to that used for string. The created object is shown in **Figure 7**.

Another object called wall maze is created using basic shape-block. The purpose of this maze system is to compare it with constrained maze and to see which one is more effective. The increase level of difficulty of maze makes it difficult for the subject to reach the end point. **Figure 8** shows the flowchart used in creation of the object. **Figure 9** shows the snapshot of third level in maze exercise. The difference between maze of **Figure 7** and **9** is that in **Figure 7** cursor is guided along the wall of the maze whereas in **Figure 9** cursor is guided to move between the walls of the maze.

4.4. Game

The virtual objects are created to simulate game like environment. The subjects can be asked to play these games. These will be a good exercise for them. In these exercises, the ball moves on screen randomly and subject has to move his hand along with it to make sure that contact is not lost. As difficulty level increases, the speed of ball increases, so, the chance of lose of contact increases and subject has to take relatively quick decision to keep touching the ball. The snap shot of created game is shown in **Figure 10**.

4.5. Graphic User Interface (GUI)

To access all the exercises from one platform, graphic user interface (GUI) has been developed. Through GUI one can launch any exercise by clicking on desired application. But before launching any application, one must make sure that previous one is closed. This is because thread used for previous application must be de-allocated

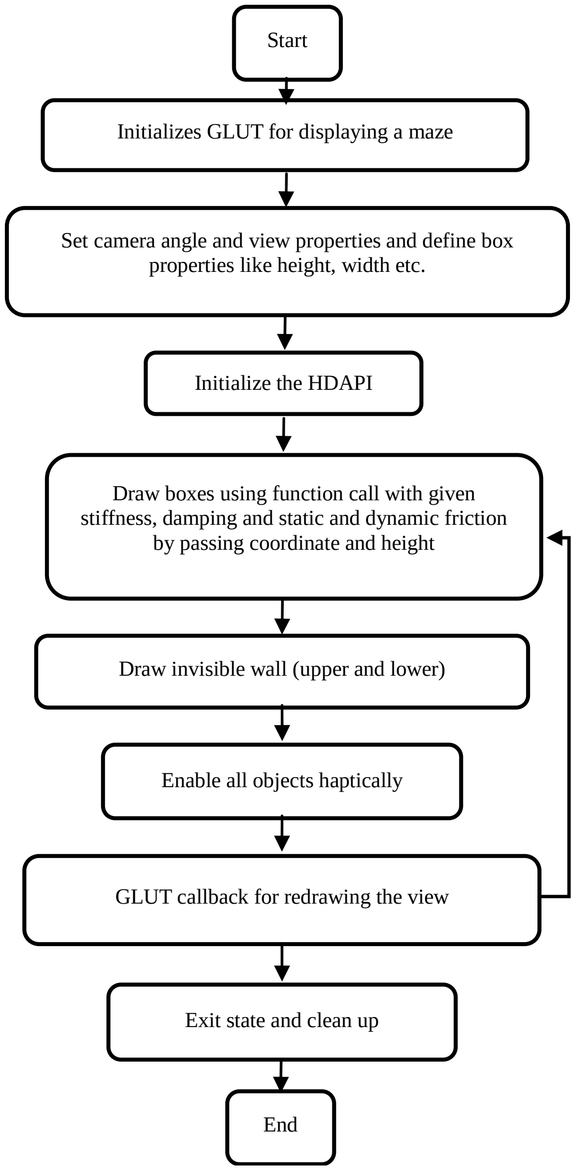


Figure 8. Flowchart used for creation of wall maze.

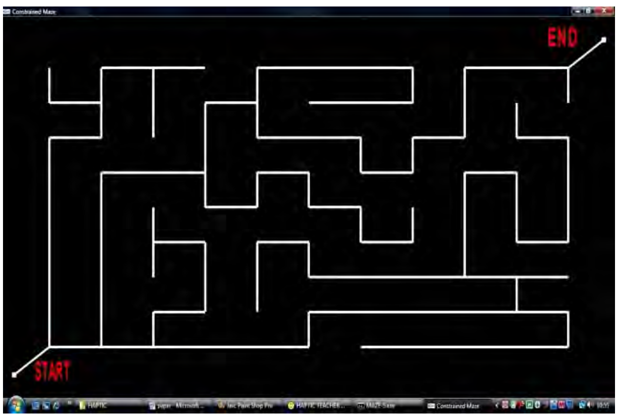


Figure 7. Constrained maze exercise.

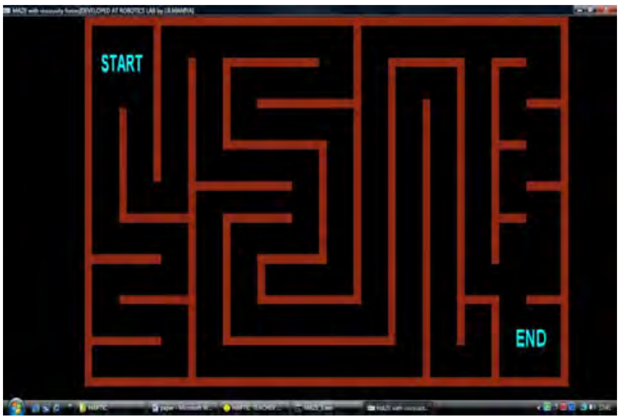


Figure 9. Wall exercise.

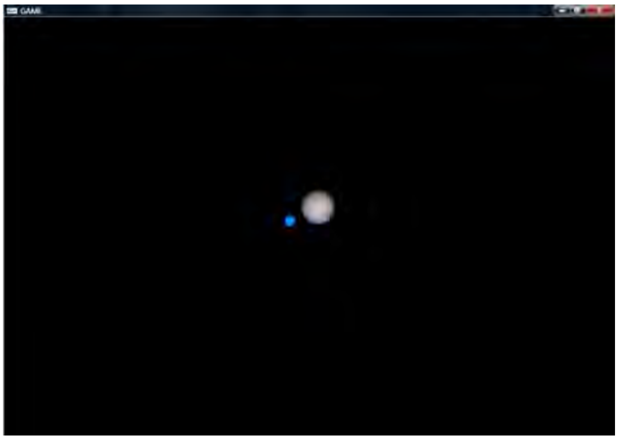


Figure 10. Game exercise.

to use it with new application. **Figure 11** shows GUI for developed system.

5. Experimental Validations on Subjects

Experiments were conducted on four subjects of age group 11-14 years as subjects from this age group were readily available. In alphabet and number exercise, they were asked to move their hand along predefined path which create letters or number using wall scenario.

Figure 12 shows time taken by one subject to complete the different letters of the alphabet. This Figure shows results for introduction, training and testing. It is seen that when the test is introduced to subject, he takes more time as compared to during training and testing. This clearly shows the improvement in writing speed of the subject. After getting practice on individual letter, subjects were given exercise to write words. The average time taken to complete different words for different subjects is shown in **Figure 13**.

Here also we see considerable improvement in the performance of the subjects in terms of speed of writing text.

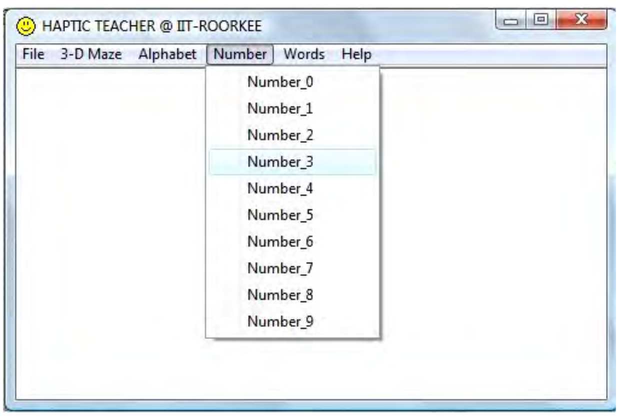


Figure 11. Graphic User Interface (GUI).

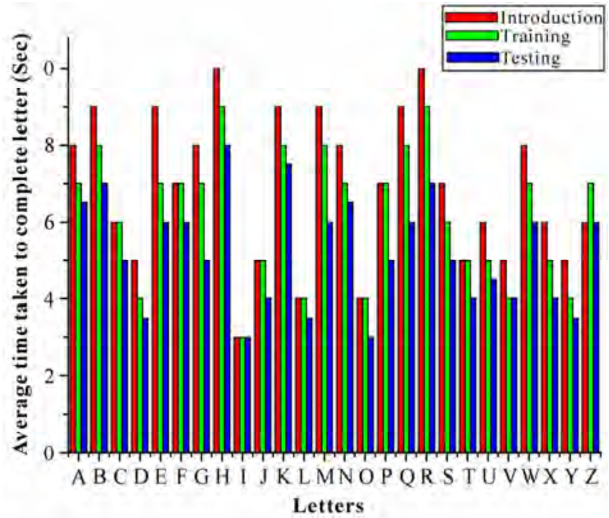


Figure 12. Time taken to complete different letters by one subject.

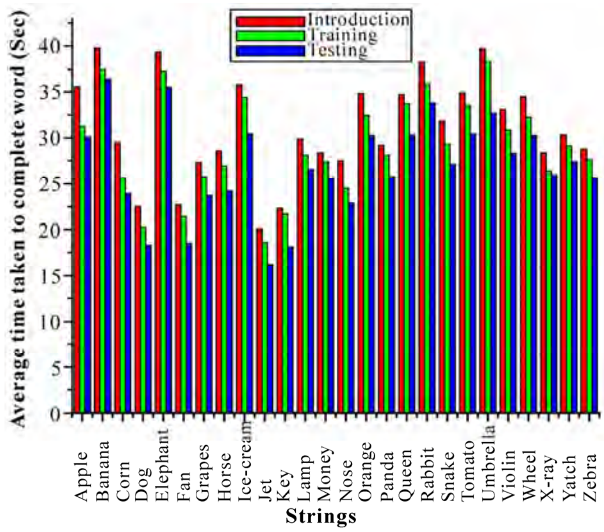


Figure 13. Average time taken by subjects to complete different words.

Figure 14 shows results for writing of numbers. Here also we see that subjects have shown improved performance in writing numbers after they got training on this system. In maze exercise, subjects had to take decision and move their hand through feasible path from many available paths to reach at end point from start point. **Figure 15** shows the performance of subjects when they performed on wall and constrained maze.

We observe from this Figure that constrained maze requires comparatively less time as compared to wall maze to complete the task of traveling from start point to end point. **Figure 16** shows performance of subjects after training on game object. We observe improvement in subject performance after getting training on designed

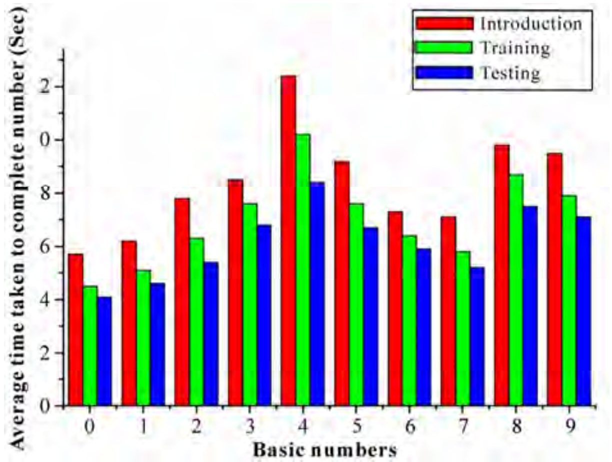


Figure 14. Average time taken by subjects to complete numbers.

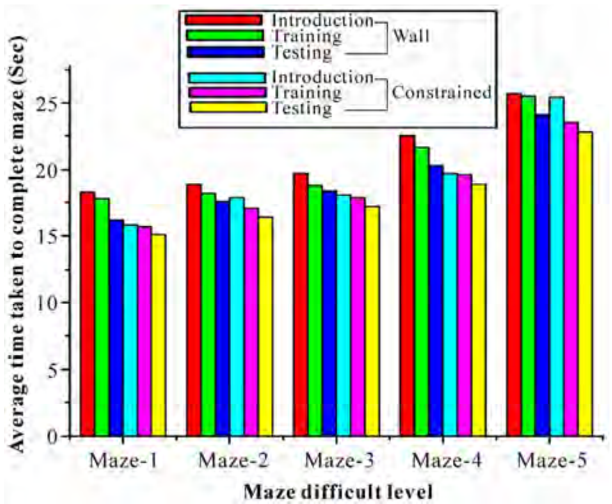


Figure 15. Average time taken by subjects to complete different maze level.

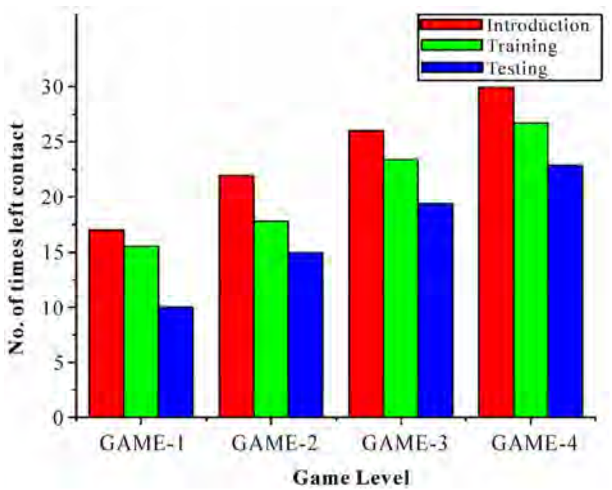


Figure 16. No. of times contact loss from ball.

system of game. As they practice more on game scenario, they are less like to lose contact with randomly moving sphere on the screen.

6. Conclusions

This work presented the idea of use of virtual objects to improve the motor skill of the patients suffering from motor disorder. To validate the idea experiments were conducted on children aged 11-14 years. It is revealed that after being introduced to system, they adapt it rapidly and their skill improved and also time taken to accomplish the task has been reduced significantly. This system has potential to train motor disordered patients and stroke patients. The improvement in performance of the children in term of speed by which they created the object is encouraging. This system can be further improved by generalizing the strings and by incorporating full sentences. The game scenario also can be added which provides good exercise.

REFERENCES

[1] D. Grossi, G. Calise, C. Correra and L. Trojano, "Selective Drawing Disorders after Right Sub cortical Stroke: A Neuropsychological Premorbid and Follow-Up Case Study," *Italian Journal of Neurological Science*, Vol. 17, No. 3, 1996, pp. 241-248.

[2] G. M. Prisco, C. A. Avizzano, M. Calcara, S. Ciancio, S. Pinna and M. Bergamasco, "A Virtual Environment with Haptic Feedback for the Treatment of Motor Dexterity Disabilities," *International Conference on Robotics and Automation*, Vol. 4, 1998, pp. 3721-3726.

[3] K. Henmi and T. Yoshikawa, "Virtual Lesson and Its Application to Virtual Calligraphy System," *IEEE International Conference on Robotics and Automation*, Vol. 2, 1998, pp. 1275-1280.

[4] R. B. Gillespie, M. S. O'Modhrain, P. Tang, D. Zaretzky and C. Pham, "The Virtual Teacher," *ASME International Mechanical Engineering Conference and Exposition*, Anaheim, CA, 1998, pp. 3354-3361.

[5] N. Pemalete, W. Yu, R. Dubey and W. Moreno, "Development of a Robotic Haptic Interface to Assist the Performance of Vocational Tasks by People with Disabilities," *IEEE International Conference on Robotics and Automation*, Washington, DC, 2002, pp. 1269-1274,

[6] C. L. Teo, E. Burdet and H. P. Lim, "A Robotic Teacher of Chinese Handwriting," 10*th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (*HAPTICS*.02), Orlando, 2002, pp. 335-341.

[7] D. Feygin, M. Keehner and F. Tendick, "Haptic Guidance: Experimental Evaluation of a Haptic Training Method for a Perceptual Motor Skill," 10*th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (*HAPTICS*.02), Orlando, 2002, pp. 40-47.

[8] N. Pernalete, S. Edwards, R. Gottipati, J. Tipple, V.

Kolipakam and R. V. Dubey, "Eye-Hand Coordination Assessment/Therapy Using a Robotic Haptic Device," *IEEE* 9*th International Conference on Rehabilitation Robotics*, Chicago, IL, USA, 2005, pp. 25-28.

[9] J. Mullins, C. Mawson and S. Nahavandi, "Haptic Handwriting Aid for Training and Rehabilitation," *IEEE International Conference on Systems, Man and Cybernetics*, USA, 2005, pp. 2690-2694.

[10] J. L. Grantner, R. Gottipati, N. Pernalete, G. A. Fodor and S. Edwards, "Intelligent Decision Support System for Eye-Hand Coordination Assessment," *The* 14*th IEEE International Conference on Fuzzy Systems*, USA, 2005, pp. 55-60.

[11] M. Mansour, M. Eid, and A. E. Saddik, "A Multimedia Handwriting Learning and Evaluation Tool," *International Workshop on Educational Multimedia and Multimedia Education*, Augsburg, Germany, 2007, pp. 103-108.

[12] C. Gouy-Pailler, S. Zijp-Rouzier, S. Vidal and D. Chêne, "A Haptic Based Interface to Ease Visually Impaired Pupils' Inclusion in Geometry Lessons," *Lecture Notes in Computer Science, Proceedings of the* 4*th international conference on Universal access in human-computer interaction: applications and services*, Beijing, China, 2007, pp. 598-606.

[13] R. Palluel-Germain, F. Bara, A. Hillairet de Boisferon, B. Hennion, P. Gouagout and E. Gentaz, "A Visuo-Haptic Device-Telemaque-Increases Kindergarten Children's Hand-writing Acquisition," *IEEE Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (*WHC*'07), Tsukaba, 2007, pp. 72-77.

[14] K. Rassmus-Grohn, C. Magnusson and H. Efiring, "AHEAD-Audio-Haptic Drawing Editor and Explorer for Education," *HAVE* 2007-*IEEE International Workshop on Haptic Audio Visual Environments and Their Applications*, Ottawa, Canada, 2007, pp. 62-66.

[15] R. Kayyali, S. Shirmohammadi, A. E. Saddik and E. Lemaire, "Daily-Life Exercises for Haptic Motor Rehabilitation," *IEEE International Workshop on Haptic Audio Visual Environments and Their Applications*, Ottawa, Canada, 2007, pp. 118-123.

[16] B. Bayart, A. Pocheville and A. Kheddar, "An Adaptive Haptic Guidance Software Module for I-TOUCH: Example Through a Handwriting Teaching Simulation and a 3D Maze," *IEEE International Workshop on Haptic Audio Visual Environments and Their Applications*, Canada, October, 2005, pp. 51-56.

J. Software Engineering & Applications, **2010, 3, 998-1004**
doi:10.4236/jsea.2010.310117 Published Online October 2010 (http://www.SciRP.org/journal/jsea)

A New Approach to Software Development Fusion Process Model

Rupinder Kaur, Jyotsna Sengupta

Department of Computer Science; Punjabi University, Patiala, India.
Email: rupadeo@gmail.com

Received August 12th, 2010; revised August 25th, 2010; accepted September 3rd, 2010.

ABSTRACT

There are several software process models that have been proposed and are based on task involved in developing and maintaining software product. The large number of software projects not meeting their expectation in terms of functionality, cost, delivery schedule and effective project management appears to be lacking. In this paper, we present a new software fusion process model, which depicts the essential phases of a software project from initiate stage until the product is retired. Fusion is component based software process model, where each component implements a problem solving model. This approach reduces the risk associated with cost and time, as these risks will be limited to a component only and ensure the overall quality of software system by considering the changing requirements of customer, risk assessment, identification, evaluation and composition of relative concerns at each phase of development process.

Keywords*: *Process Model, Fusion Process Model, Component Driven Development Approach,* 3*C-Model

1. Introduction

The importance of software process model for development of software product is well known, which include various steps that guide the team with common goals and strategies. Several software life cycle models or process models have appeared till now. All these models share certain characteristics. They identify stakeholder goal, specify key activities to be followed according to a certain sequence, work within time constraints and are based on what has been learned from past experiences.

The evolution of software process models has played a significant role in how models have diversified over time. Software development process and general solutions for organizing the software process belong to the standard themes of software engineering and have challenged theoreticians and practitioners for a long time. The causes of the software crisis were linked to the overall complexity of the software process and the relative immaturity of software engineering as a profession. The crisis manifested itself in several ways: projects running over-budge; projects running over-time; software was very inefficient; software was of low quality; software often did not meet requirements; projects were unmanageable and code difficult to maintain and software was never delivered.

The overall success in the development of software is still not achieved because each software development process or model consider only one or few concerns and specify a phase wise abstraction for the development, but no definite approach or model is specified for the phases of software process model. In current software engineering practices, ever changing requirements during the development process for large software development is still not managed by software process models. The solution space analysis concept of software engineering is very effective, but this concept is not completely integrated to software development yet. Alternative management, a technique which is used in mature engineering disciplines is not explicitly followed in software engineering discipline. The software development models till date follow fixed or iterative design and development approach. There is no scope for dynamic testing in software development process.

To make the software development effective and reliable, a new approach is required. Fusion process model is based on component driven development approach, which is different from component base software development. In Fusion process model, each component implements a problem solving model. It includes the explicit processes for technically analyzing the problem, solution space analysis, alternative management, dynamic design specification and development and scope for dynamic testing. In this paper, we present the new software process model which will address all the concerns and consider each phase of software development as software development process and provide an effective model for software development phases.

The main contributions of this paper are: a proposal of fusion based process model that will manage the concerns in software development and an integration of 3C-Model [1] in fusion process model for different phases that include the concept of Context (environment), Capture (Problem Solving concept for various development phases) and Control (based on environment and development constraints, quality criteria, mathematical and optimization techniques).

In Section 2, we discuss existing software development approaches with their shortcomings. Then we give a brief idea of problem solving model named 3C-Model in Section 3. Next, we present the fusion process model for effective and reliable software development in Section 4. Finally, we conclude and discuss future work in Section 5.

2. Related Approaches

Royce [2] proposed the first and most influential approach which is referred to as the waterfall model and has become the base for other models. In this approach, the whole process of software development is divided into separate process phases. Humphrey and Kellner [3] criticize the model by discussing the problems which are faced after implementing the model. These have linear structure and rigid design, rather than a dynamic problem-solving process, which would help in implementing the learning that result from user feedback and involvement. There was a change in software development approach with incremental and iterative models, also called phased development models. Graham [4] contrasted incremental development unfavorably with the fixed character of the waterfall model and suggested this approach to be for small systems only.

Boehm [5] proposed an approach which consists of a series of Waterfall-like cycles. Each cycle addresses the development of the software product at a further level of detail. Several papers indicate that for the development of software system, the identification of concerns, objective and alternatives is vital [6,7], and [8]. Later Boehm and Prasanta Bose [9] extended the spiral model to a variant called the Win-Win Spiral Model, also called win-win stakeholder approach to converge on a system's next level objectives, constrains and alternatives. It determines three milestones: life-cycle objectives, life-cycle architecture, and initial operational capability, which served as basis for software development process. This model has been formally specified and analyzed for consistency but only little is known about the correctness and usefulness of assumptions made during this process. The process and outcome of negotiations are not well defined [10].

Harlan Mills [11] proposed clean room approach, a quality control driven philosophy which is intended to improve the manageability and predictability of development process. This approach does not provide life cycle model, it provides specification for the software development. This approach does not provide life cycle model, it provides specification for the software development.

Alan Cline [12] paper work shows Joint Application Development (JAD) technique, which is an attempt to build collaboration process model. It is a technique for engaging a group or team of software developers, testers, customers, and prospective end-users in requirement specification and development of prototype. This is suitable for open source software development projects that rely on group email discussions among globally distributed users and developers [13]. J.Neighbors [14] laid stress on reusable software components guided by an application domain analysis, which is an alternative approach to developing software. Anton Jansen and Jan Bosch [15] research shows new perspective on software architecture that views software architecture as a composition of a set of explicit design decisions. This makes design decisions an explicit part of software architecture, which has become accepted concept in research and industry. The reuse model follows the component based approach, but this approach is not guided by domain analysis. It does not provide complete life cycle for software development because it considers only those systems which can be built using existing components only.

Rich Hilliard [16] gives an overview of the contributions of IEEE 1471 to the discipline of software architecture representation which fits in the theory of phased development model for different phases of software development model. The research done by Jonathan Lee [17] de-scribes the software engineering as problem solving process. Where the software process model approaches divide the development process into various phases/activities or according to functionality. But these models still don't follow the technique of technically analyzing the problem, where the technical problems are identified and divided into sub-problems that are first independently solved and later integrated in the overall solution. The client problems may be ill-defined and include many vague requirements, but the main focus is on precise formulation of objectives, quality criteria and the constraints for given requirement or problem. In technical analysis part, we can easily put this specification on each small unit of problem.

Providing a solution for a given problem is not simple, it involves the accumulation and use of huge amount of knowledge. The solution space analysis approach is still not integrated into software process models. It aims to identify the right solution domains for the given problems and extract the relevant knowledge from these domains to come up with a feasible solution. To provide quality software, it is necessary to identify the important knowledge sources for a given problem. Not all the solutions identified for a given problem are desirable. In the alternative management process, different alternative solutions are searched and evaluated against explicit quality criteria [17,18]. The high risk in software development led to the inclusion of managerial, financial and psychological factors in models [19,20], and [21]. Shaw and Garlan [22] identify seven levels of design specification capability which supports the concept of components, composition, validation, alternatives and finally automation. In the component based development, cost, time and reliability risk for an organization developing software system will shrink to component level that can be managed effectively at any stage.

Thus, a number of software process models have been studied. Most of the existing techniques manage one or more concerns of development process. From the insight gained from this study, contemporary software process models is needed, which handles various issues like requirement changes, software reuse, flexible design, user involvement and tight control over quality, cost and schedule can be overcome.

3. 3C Model

The 3C Model helps in generalizing the software development process in which a problem specification is transformed to a solution by decomposing the problem into sub-problems that are independently solved and integrated into an overall solution. This consists of multiple cycles; each cycle in 3C-Model corresponds to a transformation from one state to another, consisting of a problem specification state and a design state. The problem specification state defines the set of problems that still needs to be solved. The design state represents the tentative design solution that has been lastly defined. Initially, the design state is empty and the problem specification state includes the initial requirements. After each state transformation, a sub-problem is solved. In addition a new sub-problem may be added to the problem specification state. Each transformation process involves an evaluation step whereby it is evaluated whether the design solutions so far (design state) are consistent with the initial requirements and if there are any additional requirements identified during the evaluation. In particular, 3C-Process includes an explicit phase for searching design alternatives in the corresponding solution space and selecting these alternatives based on explicit quality criteria.

4. Fusion Process Model

Fusion is component driven software process model, where each phase implements a problem solving model. These phases address what is to be built, how it will be built, building it and making it high quality. The problem solving model includes the explicit processes for technically analyzing the problem, solution space analysis, alternative analysis, dynamic design and development and scope for dynamic testing. With the problem analysis process, technical problems are identified and structured into loosely coupled sub-problems that are first independently solved and later integrated in the overall solution. In the solution space analysis process, requirements are specified using any representation and this should be refined along the software development process until the final software is delivered. In the alternative analysis process, different alternative solutions are searched and evaluated against explicit quality criteria. Dynamic design and development is component base approach, which provides scope for dynamic changes during the development life cycle. As fusion process follows the component design approach, it provides scope for dynamic testing (component base testing).

3C-Model assist fusion process model in generalizing the process of solving the problems in each phase. It implements component driven development approach, which provides a dynamic nature to complete software development. This makes the software development scope wider and provides firmer control over software development process. Because of the component driven approach, the risk associated with cost and time will be limited to component only and ensure the overall quality of software system, reduce the development cost and time by considering the changing requirements of customer, risk assessment, identification, evaluation and composition of relative concerns at each phase of development process. There are five fundamental phases in fusion process model and one fusion process controller to control and co-ordinate the overall development process, as shown in **Figure 1**.

4.1. Project Preparation

The project preparation phase provides the initial planning and preparation for software development project. Although each project has its own unique objectives, scope, and priorities, this phase assists in identifying and planning the primary focus areas that need to be considered. These include technical as well as project management

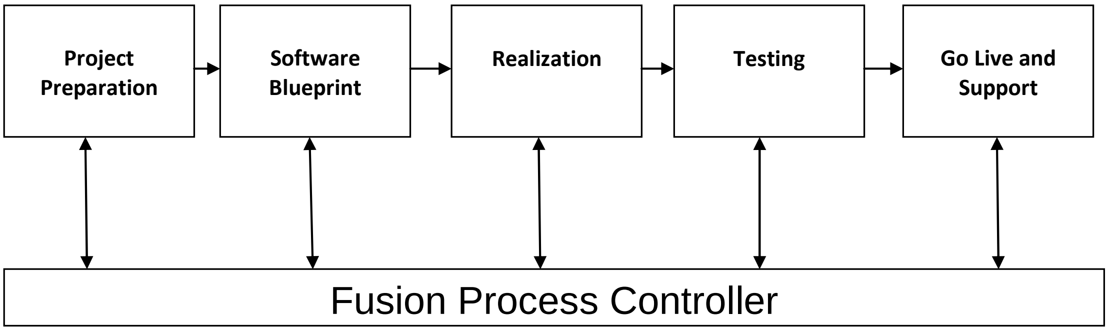


Figure 1. Fusion process model.

issues. Addressing these issues early in development will ensures that the project will proceed efficiently and establish a firm foundation for a successful development. While developing a software project, requirements definition is often considered as a one-time activity. In fact it starts with initiation of the project and is on-going activity. In feasibility analysis, requirements definition plays an important role. Every project has a feasibility analysis, regardless of the methodology used. It is essential to include important requirements definitions in feasibility analysis. When done poorly, as so often happens, the project is almost certainly destined to fail. Feasibility analysis is often referred as "project initiation" whether or not to do a project. Essentially software project team identifies what they expect the project to produce and whether it seems worthwhile to do so. If a project exists, team has made decision about it.

Extracting requirements of a desired software product is the first task in creating it. This process is called requirements elicitation. While customers probably believe they know what the software should do, it may require skill and experience in software engineering to recognize incomplete, ambiguous or contradictory requirements. Requirement analysis process provides an understanding of the client perspective of the software system. After requirements elicitation, client requirements are mapped to technical problems in the technical problem analysis process. The problem analysis process consists of the following steps:

- Generalize the Requirements: whereby the requirements are abstracted and generalized.
- Identify the Sub-Problems: whereby technical problems are identified from the generalized requirements.
- Specify the Sub-Problems: whereby the overall technical problem is decomposed into sub-problems.
- Prioritize the Sub-Problems: whereby the identified technical problems are prioritized before they are processed.

Problem reduction is a strategic approach to manage complexity. A widely known method for solving large and complex problems is to split them into simpler problems and then iteratively apply this process. The process is put into action until the sub-problems are reduced to a level of complexity at which they are easily solved or at least exhibit an irreducible level of difficulty. This paradigm for solving problems is called problem reduction. In this, a problem in a given domain is decomposed into a structured set of sub-problems in the same domain. Each sub-problem is evaluated for suitability to be further decomposed until each sub-problem is determined solvable. This problem reduction paradigm has been successfully applied to problems in a variety of application domains and in many phases of the process in which a top-down decision making strategy is applied.

The problem reduction can be expensive if not handled properly. Often, the same process must be done repeatedly for a similar type of problem with only minor differences. As a result, problem reduction may cost even more over time as problems become more complex. An important approach for handling the side effects of problem reduction is to build reusable sub-problems and solutions, instead of continually reinventing a related system reductive hierarchy. Such reusable sub-problems and solutions can be stored in a components library and retrieved as required. Complete solutions can then be obtained by using and reassembling appropriate sub-solution components.

4.2. Software Design

Architecture is established in the design phase. This phase starts with the inputs delivered by the initial phase and maps the requirements into architecture. The architecture defines the components, their interfaces and behaviors. The deliverable design document is the architecture. The design document describes a plan to implement the requirements. This phase represents the "how" phase. Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established. The design may include the usage of existing components.

The Solution Domain Analysis process applied in software design phase aims to provide a solution domain model that will be utilized to extract the architecture de-sign solution. It consists of the following activities:

- Identify and prioritize the solution domains for each sub-problem
- Identify and prioritize knowledge sources for each solution domain.
- Extract solution domain concepts from solution domain knowledge.
- Structure the solution domain concepts.
- Refine the solution domain concepts.

4.2.1. Identify and Prioritize the Solution Domains

To the overall problem and each sub-problem, search for the solution domains are prepared that provide the solution abstractions to solve the technical problem. The solution domains for the overall problem are more general than the solution domains for the sub-problems. In addition, each sub-problem may be recursively structured into sub-problems requiring more concrete solution domains.

An obstacle in the search for solution domains is possibly the large space of solution domains which leads to a time-consuming search process. To support this process, categorizations of the solution domain knowledge into smaller sub-domains had been executed. There are different categorization possibilities. The solution domain knowledge can be categorized into application, mathematical and computer science domain knowledge etc. The application domain knowledge refers to the solution domain knowledge that defines the nature of the application, such as reservation applications, banking applications, control systems etc. Mathematical solution domain knowledge refers to mathematical knowledge such as logic, quantification, calculation and optimization techniques, etc. Computer science domain refers to knowledge of the computer science solution abstractions, such as programming languages, operating systems, databases, analysis and design methods etc.

4.2.2. Identify and Prioritize Knowledge Sources

Each identified solution domain covers a wide range of solution domain knowledge sources. These knowledge sources may not all be suitable and vary in quality. For distinguishing and validating the solution domain knowledge sources we basically consider the quality factors of objectivity and relevancy. The objectivity quality factor refers to the solution domain knowledge sources itself, and defines the general acceptance of the knowledge source. The relevancy factor refers to the relevancy of the solution domain knowledge for solving the identified technical problem.

4.2.3. Extract Solution Domain Concepts from Solution Domain Knowledge

Once the solution domains have been identified and prioritized, the knowledge acquisition from the solution domain sources can be initiated. The solution domain knowledge may include a lot of knowledge that is covered by books, research papers, case studies, reference manuals, existing prototypes/systems etc. Due to the large size of the solution domain knowledge, the knowledge acquisition process can be a labor-intensive activity, so a systematic approach for knowledge acquisition is required.

In this approach, we make a distinction between the knowledge elicitation and concept formation process. Knowledge elicitation focuses on extracting the knowledge and verifying the correctness and consistency of the extracted data. Hereby, the irrelevant data is disregarded and the relevant data is provided as input for the concept formation process. Knowledge elicitation techniques are eminent and its role in the knowledge acquisition process is reasonably well-understood. The concept formation process is mapping the extracted knowledge with technical problems. The concept formation process utilizes the knowledge and get abstract to form concept. One of the basic abstraction techniques in forming concepts is by identifying the variations and commonalities of extracting information from the knowledge sources.

4.2.4. Structure the Solution Domain Concept

The identified solution domain concepts are structured using parent-child relationship. Here all the attributes and operations associated with the concept are defined.

4.2.5. Refinement of Solution Domain Concepts

After identifying the top-level conceptual architecture, the focus is on each sub-problem and follows the same process. The refinement may be necessary if the architectural concepts have a complex structure themselves and this structure is of importance for the eventual system.

The ordering of the refinement process is determined by the ordering of the problems with respect to their previously determined priorities. Architectural concepts that represent problems with higher priorities are handled first and in the similar manner the refinement of the architectural concepts is done.

4.3. Realization

The purpose of realization phase is to develop software system for requirements based on the software design;

the team builds the components either from scratch or by composition. Given the architecture document from the design phase and the requirement document from the analysis phase, the team builds exactly what has been requested, though there is still room for innovation and flexibility.

4.3.1. Alternative Design Space Analysis

The alternative space is define as a set of possible design solutions that can be derived from a given conceptual software architecture. The alternative design space analysis aims to depict this space and consists of the two sub-processes: define the alternatives for each concept and describe the constraints. Let us now explain these sub-processes in more detail.

4.3.1.1. Define the Alternatives for each Concept

In this approach the various architecture design alternatives are derived from well-established concepts in the solution domain that have been leveraged to the identified technical problems.

4.3.1.2. Describe the Constraints

The total set of alternatives per concept may be too large and/or not relevant for solving the identified problems. Therefore, to define the boundaries of the architecture it is necessary to identify the relevant alternatives and omit the irrelevant ones.

4.4. Testing

Quality of software product is very important while developing it. Many companies have not learned that quality is important and deliver more claimed functionality but at a lower quality level. It is much easier to explain to a customer why there is a missing feature than to explain to a customer why the product lacks quality. A customer satisfied with the quality of a product will remain loyal and wait for new functionality in the next version. Quality is a distinguishing attribute of a system indicating the degree of excellence.

In many software engineering methodologies, the testing phase is a separate phase which is performed by a different team after the implementation is complete. There is merit in this approach; it is hard to see one's own mistakes, and a fresh eye can discover obvious errors much faster than the person who has read and re-read the material many times. Unfortunately, delegating testing to another team leads to a slack attitude regarding quality of the implementation team.

Alternatively, another approach is to delegate testing in the whole organization. If the teams are to be known as craftsmen, then the teams should be responsible for establishing high quality across all phases. Sometimes, an attitude change must take place to guarantee quality. The testing technique is from the perspective of the system provider. Because it is nearly impossible to duplicate every possible customer's environment and because systems are released with yet-to-be-discovered errors, the customer plays an important, though reluctant, role in testing.

4.5. Go Live and Support

The purpose of the Go Live and Support phase is to cut over to live productive operation and to continuously support and improve live operations. There are two distinct periods of this phase:

4.5.1. Project End

During the time when the system is first live, all issues and problems are resolved, transition to the production support team is finalized, knowledge transfer is completed, and the project is signed off.

4.5.2. Continuous Improvement

Now that the project is over, the production support team monitors the system and resolves live business process issues. Proper change management procedures are established, and ongoing end-user training is conducted. Plans are made to continuously review and improve business processes.

4.6. Fusion Process Controller

The controller part is not a phase in process model, but it is integral part of fusion process model. The controller part helps to achieve the component driven approach by listing the details of components which are added due to requirement changes or because of new requirements. By implementing Fusion Process Controller the current software development process will not be affected by changes required due to new requirements or modifications. The affected components can be taken care separately till these components matches with the current development process.

5. Conclusions

We have presented a Fusion Process Model for software development process and discussed the concept of 3C-Model for each phase of development process model. In this approach transformation of a problem specification to a solution is made by decomposing the problem into sub-problems that are independently solved and integrated into an overall solution. This process consist of multiple cycles, were each cycle transform from problem specification state to design state. After each state trans-formation, a sub-problem is solved and a new sub-problem possibly be added to the problem specification state. Every transformation process engages an evaluation step, evaluation of design state of the initial requirements is done and verifies if additional requirements identified during this step. In particular, this process includes an explicit phase for searching design alternatives in the corresponding solution space and selecting these alternatives based on explicit quality criteria. Our work has shown that how this approach helps in controlling the overall development process by implementing component based approach. Since it is component driven approach, the threat tied to cost and time will be restricted to component only, ensuring the overall quality of software product, considering the changing requirements of customer, risk assessment, identification, evaluation and composition of relative concerns at each phase of development process.

REFERENCES

[1] R. Kaur and J. Sengupta, "Development and Analysis of 3C-Model for Software Development Lifecycle," *IEEE* 2*nd International Conference on Computer Engineering and Technology* (*ICCET* 2010), 16-18 April 2010, Chengdu, China, pp. 688-691.

[2] W. W. Royce, "Managing the Development of Large Software System," *Proceedings* 9*th International Conference on Software Engineering*, *IEEE Computer Society*, USA, 1987, pp. 328-338.

[3] W. S. Humphrey and M. I. Kellner, "Software Process Modeling: Principles of Entity Process Models," *Technical Report*, ACM Press, New York, February 1989, pp. 331-342.

[4] D. Graham, "Incremental Development and Delivery for Large Software System," *IEEE Computers*, Vol. 25, No. 11, 1992, pp. 1-9.

[5] B. Boehm, "Software Engineering Economics," *IEEE Transaction on Software Engineering*, Vol. 10, No. 1, 1984, pp. 4-21.

[6] B. Boehm, "Anchoring the Software Process," *IEEE Transaction on Software Engineering*, Vol. 13, No. 4, 1996, pp. 79-82.

[7] B. Boehm, "A Sprial Model of Software Development and Enhancement," *IEEE Computer*, Vol. 21, No. 5, 1988, pp. 61-72.

[8] B. Boehm and D. Port, "Escaping the Software Tar Pit: Model Clashes and How to Avoid Them," *Software Engineering Note*, Vol. 24, No. 1, 1999, pp. 36-48.

[9] B. Boehm and P. Bose, "A Collaborative Spiral Software Process Model Based on Theory W. Processing of ICSP," IEEE Press, New York, 1994.

[10] A. Egyed and B. Boehm, "Analysis of System Requirements Negotiation Behavior Patterns," 7*th Annual International Symposium Engineering*, USA, 1997, pp. 269-276.

[11] F. P. Deek, J. A. M. McHugh and O. M. Eljabiri, "Strategic Software Engineering: An Interdisciplinary Approach," Auerbach Publications, USA, 2005, pp. 31-35.

[12] A. Cline, "Joint Application Development (JAD)," 2010. http://www.carolla.com/ wp-jad.htm

[13] W. Scacchi, "Process Models in Software Engineering," Institute for Software Research, University of California, Irvine, October 2001.

[14] J. Neighbors, "The Draco Approach to Constructing Software from Reusable Components," *IEEE Transaction on Software Engineering*, Vol. 10, No. 5, 1984, pp. 564-574.

[15] A. Jansen and J. Bosch, "Software Architecture as a Set of Architecture Design Decision," *IEEE Computer Society*, 2005, pp. 109-120.

[16] R. Hilliard, "IEEE Std. 1471 and Beyond," January 2001, pp. 1-3.

[17] J. Lee, "Software Engineering with Computational Intelligence," Springer Publication, New York, 2003, pp. 183-191,

[18] X. Ferre and S. Vegas, "An Evaluation of Domain Analysis Methods," 4*th CASE/IFIP*8 *International Workshop in Evaluation of Modeling in System Analysis and Design*, 1999, pp. 2-6.

[19] A. Hamid and Madnick, "Lesson Learned from Modeling the Dynamics of Software Development," *Communication ACM*, Vol. 32, No. 12, 1989, pp. 14-26.

[20] J. Ropponen and Lyytinen, "Components of Software Development Risk: How to Address Them?" *A Project Manager Survey*, *IEEE Transaction on Software Engineering*, Vol. 26, No. 2, 2000, pp. 98-112.

[21] B. Boehm, "Software Engineering Economics," *IEEE Transaction on Software Engineering*, Vol. 10, No. 1, 1984, pp. 4-21.

[22] N. Medvidovic and R. M. Taylor, "A Classification and Comparison Framework from Software Architecture Description Languages," *IEEE Transactions on Software Engineering*, Vol. 26, No. 1, 2000, pp. 70-93.

J. Software Engineering and Applications, **2010, 3, 901-1004.**
Published Online October 2010 in SciRes (http://www.SciRP.org/journal/jsea).

TABLE OF CONTENTS

Volume 3 Number 10 October 2010

