RIGA TECHNICAL UNIVERSITY
Faculty of Computer Science and Informatin Technology
Institute of Information Technology

Pāvels OSIPOVS

Doctoral student of program "Information Technology"

THE USE OF PERSONAL ADAPTIVE BEHAVIOR PROFILE FOR DETECTING ANOMALOUS ACTIVITY OF ELECTRONIC INFORMATION SYSTEM USER

Doctoral thesis

Scientific supervisor
Dr. habil. sc. comp., professor
A. BORISOVS

RTU Press

Rīga 2015

This work has been supported by the European Social Fund within the project «Support for the implementation of doctoral studies at Riga Technical University».

RIGA TECHNICAL UNIVERSITY
FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
Institute of Information Technology

THE USE OF PERSONAL ADAPTIVE BEHAVIOR PROFILE FOR DETECTING ANOMALOUS ACTIVITY OF ELECTRONIC INFORMATION SYSTEM USER

Pāvels Osipovs

Annotation

This research focuses on the problem of anomaly level calculation in the behavior of the user of an electronic information system. This type of problem is important for a variety of electronic systems that provide access to financial, medical, military and similar sensitive data. Such systems are well protected against external threats; however monitoring of internal, already authorized users, is less effective at the moment.

To solve the problem, an approach is implemented that allows specifics of interests of the users to be represented as a graph of the Markov chain. Having this self−learning Behavioral Profile of the User (UBP), it is possible to calculate the level of abnormality of each its request to the system. A method for improving the accuracy of calculating the level of anomalous behavior through using a Personal Adaptive UBP (PAUBP) is developed. A method that uses a dynamic threshold for the anomaly level in order to increase the efficiency of classification between the "*normal*" and "*abnormal*" behavior is elaborated. A method of estimating the possibility of introducing the approach developed into the information systems having different structures is established. An approach to evaluate the effectiveness of UBP is proposed. A software system that enables making experiments with all the required characteristics of UBP is created. The experimental results show that PAUBP is more effective in solving the considered problem than the general UBP.

The work contains 144 p., 21 tables, 91 figures and 108 references.

RĪGAS TEHNISKĀ UNIVERSITĀTE
DATORZINĀTNES UN INFORMĀCIJAS TEHNOLOĢIJAS FAKULTĀTE
Informācijas tehnoloģijas institūts

ELEKTRONISKĀS INFORMĀCIJAS SISTĒMAS LIETOTĀJU PERSONĪGĀ ADAPTĪVĀ PROFILA PIELIETOŠANA ANOMĀLAS UZVEDĪBAS ATKLAŠANAI

Pāvels Osipovs

Anotācija

Darbs tiek veltīts elektroniskās informācijas sistēmas lietotāja uzvedības anormalitātes līmeņa izskaitļošanas problēmai. Šāda veida uzdevumi ir aktuāli dažādās elektroniskās sistēmās, kuras nodrošina piekļuves iespēju finansiāliem, medicīniskiem, kara un citiem sensitīviem datiem. Tādas sistēmas ir ļoti labi aizsargātas no ārējiem draudiem, bet iekšējo, jau autorizēto lietotāju monitorings pašreiz nav tik efektīvs.

Uzdevuma risināšanai darbā tika realizēta pieeja, kas ļāva lietotāju interešu īpašības attēlot Markova ķēdes grafa veidā. Iegūstot šādu pašapmācošu lietotāja uzvedības profilu (UP), rodas iespēja izskaitļot katra lietotāja sistēmai nodotā vaicājuma anomalitātes līmeni. Pielietojot personālo adaptīvo uzvedības profilu (PAUP), izstrādāta metode, kas ļauj paaugstināt uzvedības anomalitātes līmeņa izskaitļošanas precizitāti. Lai paaugstinātu lietotāju klasifikācijas uz „*normālajiem*" un „*anomālajiem*" efektivitāti, izstrādāta metode, kas izmanto dinamisku anomalitātes līmeņa slieksni.

Tika izstrādāta uzvedības profilu efektivitātes vērtēšanas metodika, kas paredzēta profiliem, kas izveidoti gan izmantojot darbā izstrādātās pieejas, gan izmantojot citas metodes. Izpildīts liels eksperimentu klāsts, izskatot dažādus veidojamo profilu raksturojumus; novērtēts vaicājumu anomalitātes līmeņa izskaitļošanas ātrums, apmācošu un analizējamu datu raksturlielumu ietekme uz klasifikācijas precizitāti utt. Izstrādāts programmnodrošinājums, kas ļāva eksperimentēt ar datiem par uzvedību un ar pašiem UP, mainot visus nepieciešamos raksturlielumus. Pēc eksperimentu rezultātiem var secināt, ka, risinot uzstādīto uzdevumu, PAUP ir efektīvāks nekā vispārīgs UP.

Darba apjoms − 144 lpp., 21 tabula, 91 attēli. Darbā ir atsauces uz 108 avotiem.

TABLE OF CONTENTS

INTRODUCTION

The detection of anomalous activity of users within the framework of infrastructure of modern difficult information systems is a topical task. Financial, reputation and information losses due to loss of important information in large companies can be significant.

A study of level of losses of large international companies from different cyber-attacks [18] has been performed in 2013 and showed that losses of one company can achieve 50 million dollars. Larger companies have more possible damages. The reason of losses is not only directly stolen money or data, but also expensive procedures on detection and elimination of weak spots in the compromised information systems.

In accordance with classification of attacks [82] there are four layers of intrusions on information infrastructure of companies: OSI [23] *network layer*, *data layer, operating system layer* and *application layer* [25, 82]. The specifics of attacks on each layer differs significantly, so the methods of their detection at each layer have to be as much as possible adjusted under the features of its realization. According to statistics of number of attacks, methods to detect attacks are first concentrated at network layer. However if by frequency of attacks network layer is most widespread, by damage because of successful attacks − the leader is application layer, when access to data is provided by information system. Different sources [18, 25] show that 30−90% of business users consider that exactly internal attacks (Malicious insiders) cause the most significant loss. Besides, it is quite difficult to detect such attacks, because in most cases they are detected already after execution, post factum.

The essence of the attacks performed by internal users is that under the registration record of legitimate user the system is used by other person. For the system, a registration record corresponds to the user – the appropriate person. Confirming personality during procedure of authorization/ authentication, the user gets all rights for used registration record (Fig. 0.1).

As most of security systems are oriented towards prevention of breaking the system from outside, the activity of the authorized users is not analyzed in detail. Such method is determined by confidence in reliability of the used perimeter protection measures, as well as by technical limitation on monitoring of every query from all users of the system in real−time detection mode.

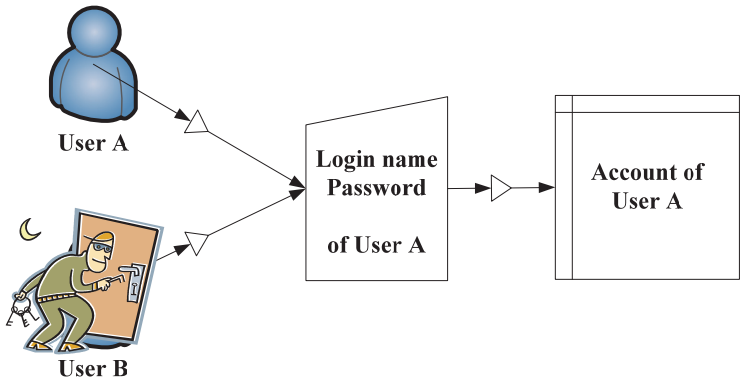


Fig. 0.1 User authentication for the system

Typical safety systems (Fig. 0.2) provide a good protection against external intrusions. Methods of successful counteraction exist for a long time, for example, counteraction to often− used intrusion techniques such as scan−out ports, refusing service, search for password in the automatic mode with a dictionary, usage of non−standard values in the packages of different levels, etc. However, activity of the user that has already been authorized is not controlled as strictly as queries that come from outside.

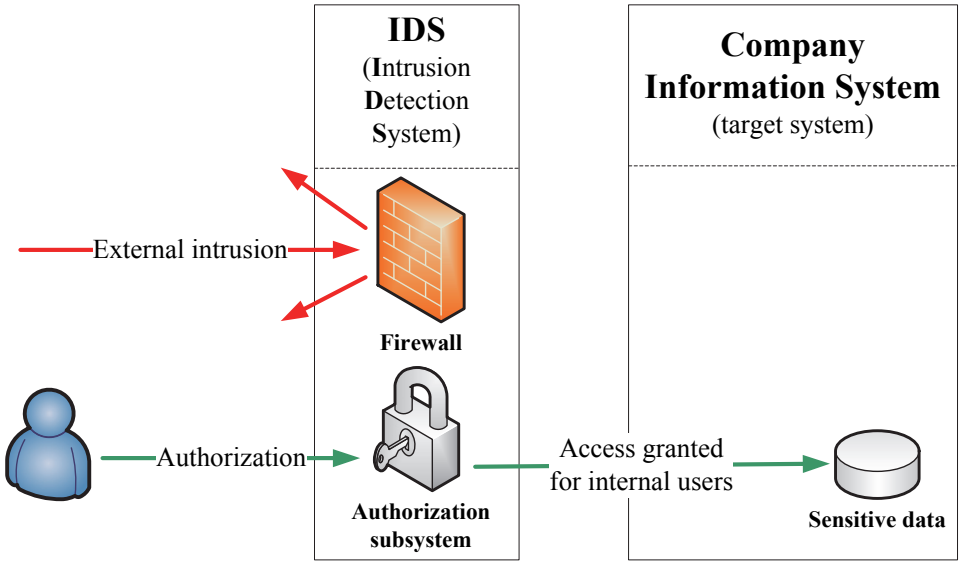


Fig. 0.2 Typical approach for security procedure

For detection of attacks from internal users the fact can be used that in spite of the user has the right of access to all sources of data in the system within the framework of his professional activity, in most cases he uses only a certain number of available services. Such situation is described on Fig. 0.3, where level of thickness of line shows the degree of interest of user in corresponding service. Thus, frequency of access to each source also depends on

belonging of person to a certain professional group of users, as well as his personal features and preferences. Therefore, having a profile of typical interests of user during previous sessions in the system, it is possible to analyze his current behavior in the presence of significant differences – anomalous behavior.

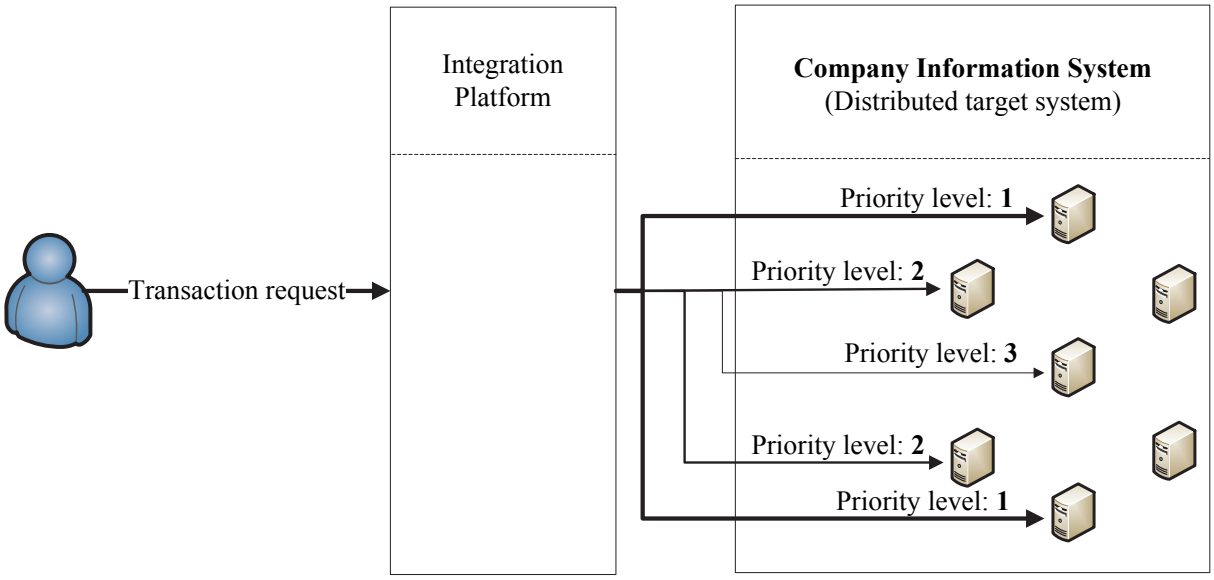


Fig. 0.3 User interest priorities to the services presented by target system

Most researches and real applications in the field of detection of attacks apply a descriptive method, when every Intrusion Detection System (IDS) uses the base of typical sets of intrusion signs and detects only such attacks. However, this method is not appropriate against new types of attacks, as well as in case of non-legitimate usage of the system by already authorized user. Usually the number of such anomalous queries does not exceed 2−3% of total number. To increase efficiency of fight against such problems statistical methods are used. Some methods are based on the construction of user's "*normal*" behavior model.

Theme topicality

The detection of anomalous behavior of electronic information system user requires complex method for its solving. Existing decisions focus on monitoring anomalies of separate components of the target system. Creating a complex approach that provides protection at all levels of functioning of complex information system is a difficult theoretical and technical

problem. The **base approach** described in this research is based on the use of Markov chains and allows obtaining effective and fast (by using the Markov property of the lack of memory) classifier of the user request level of the anomaly. Further increase of its calculation rate is topical because requirements to user queries processing time constantly increase. Methods for improving the efficiency of basic approach are also developed.

Research objective and tasks

The main objective of this research is to develop and study possibilities how to increase efficiency of detection of the user's anomaly behavior, due to usage of behavior graph adjusted to specifics of the appropriate person behavior and taking into consideration initial set of requirements.

As a basic research method, it is necessary to develop a methodology of realization of experiments that includes the development of experimental system allowing testing different descriptions of made profiles. Based on developed methodology it is also necessary to realize experimental system, using the program.

To achieve the set aim, taking into account the existing limitations, it is necessary to implement the following tasks:

- to investigate the existing approaches to formalization of user behavior and their use for anomalous activity detection;
- to investigate possibilities on realization of User Behavior Profile and Personal Adaptive Profile of User Behavior;
- to develop methodology for comparing User Behavior Profile and Personal Adaptive Profile of User Behavior and using it to implement a comparative analysis;
- to develop experimental software environment and to conduct experiments for comparing efficiency of anomalous behavior detection using both approaches. If possible, to show that Personal Adaptive Profile of User Behavior in general will be the best classifier of anomalous behavior;
- to develop methodology for introduction the developed approach to modern information systems of different structure.

Requirements to the system

At present, more and more state, municipal and private organizations operate a business on the Internet. Existent IDS provide sufficient level of protecting from external intrusions, but detection of anomalous activity of internal users is not at a high level. Data safety requirements available through these systems frequently are very strict, so the system used for detection of such type of activity has to provide certain quality of service, complying with time limitations of every transaction analysis, amount of used equipment, and programs, and service response time.

The main requirements to the system for the level anomaly calculation are the following:

- possibility of system self−learning, without the involvement of experts;
- minimum amount of information from the target system is required for analysis;
- maximum level of isolation between the created system and the target system;
- calculation speed of the level of request abnormality must allow doing it in real time.

Task setting

The nature of user behavior is not determined and is difficult to formalize ies, but within the framework of limitation of the number of possible actions, imposed by the target information system it is possible to develop behavior profile describing the features of the activity of the appropriate person or the group of users unified by common professional interests.

The main task of this research is to develop effective method for determination of the anomaly level of user behavior. Realization of basic approach is considered, as well as methods for increasing its efficiency. The task of theoretical development and program realization of experimental version of such system is set. Experimental version is used for the receipt of data about behavior of the offered approach within the framework of both artificial and real data about user behavior. Summarized description of used method is shown on Fig. 0.4.

The anomaly level is calculated for every query produced by user within the framework of work with the target system. The transaction inquired by the user is analyzed on level of anomaly of such query by current user. Level of anomaly is calculated based on the

history of previous application of the system. As a result, the less typical query is, the higher its level of anomaly will be.

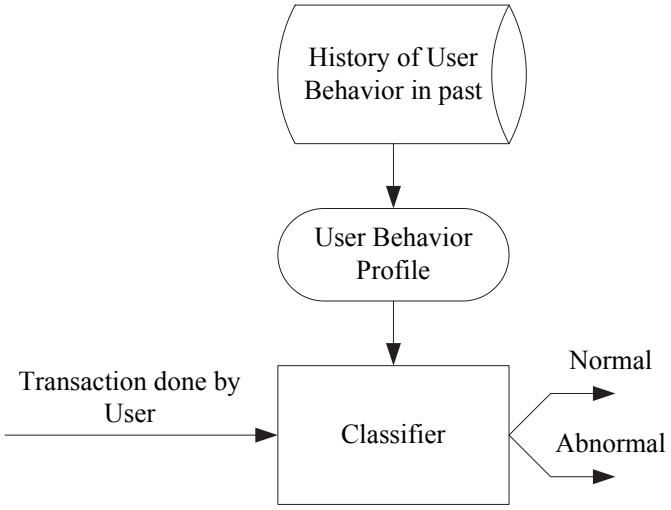


Fig. 0.4 Typical method for solving intrusion detection task

Research hypotheses

Basic hypothesis set in this thesis is user behavior profile based only on data about his personal interests within the framework of the system will most effectively calculate the anomaly level in his (user) behavior.

Within the framework of basic hypothesis, the following lower-level hypotheses are set:

- the use of the graph of Markov chain is the rapid and effective method of formalization user behavior;
- there is the possibility to compare efficiency of user behavior profiles;
- behavior of different users of the target system leads to different behavior profiles.

The first hypothesis supposes fundamental possibility of presenting data about characteristics of user behavior as a weighted oriented graph having Markov property. If it is not correct, other research has no meaning. Calculating efficiency of such presentation is also important, because even in case of fundamental possibility, if significant time and calculating resources are required for this purpose, the use of Markov chains in this task is irrational.

The second hypothesis − in case if it is not correct, there is no possibility to compare the efficiency of behavior profiles realized, using different approaches, which does not allow implementing further comparative analysis.

The third hypothesis – possibility to create experimental software system is based on the statement that the third hypothesis is correct. As real information system data are not available, it is necessary to enable generation of artificial data about behavior, whose structure is similar to real data.

The proof of each of the above−mentioned hypotheses will mean that a basic hypothesis is correct.

Scientific novelty

The main achievements described in the dissertation that meet the criteria of scientific novelty are as follows:

- As a more efficient classifier, anomalous behavior a method is proposed that takes into account the behavior of each user of the target information system.
- A method for dynamic changing the threshold for the level of abnormality is proposed to enable more accurate operation of the final classifier type of behavior.
- A method is proposed for the generation of synthetic data containing different priorities of user's interests by the use of the exponential distribution with different values of its parameters. In addition, the possibility of using other types of distributions is pointed out.
- A technique is proposed for comparing the effectiveness of behavior profiles using different approaches trained on data having a variety of characteristics and containing different values of internal variables.
- Estimates of the impact of the internal parameters of the profile behavior on the accuracy of classification of behavior type are obtained.

Practical significance

Fundamental difference of the researched type of attacks from other methods of intrusions is that after successful passing of all authorization and authentication procedures,

the user has an access to plenty of different services, but the specifics of his professional activity increases the frequency of usage of those services, which are used more frequently. Therefore, the task is to detect atypical usage of the system services, in case if they are in principle available for usage.

At present, the threat of internal attacks is important among various electronic information systems. The results obtained can be realized in various information systems:

- different public and private institutions (medical institutions, credit organizations, e-libraries, archives, local governments, political and public organizations, police, military, tax and bank authorities) have access to the information that has to be limited;
- it is possible to create a profile of typical owner of portable devices, for example – mobile phones. In this case, features of usage of device will be an analogue of person's digital signature;
- the considered method can be used not only to construct behavior profile, but also in other tasks, where sets of data determine the specifics of their author. For example, construction of author's writing style based on his works;
- the approach developed for generation of artificial data about the use of the system by a person with various priorities of interests can also be used for the development of different software systems.

Structure of the thesis

The first chapter is devoted to the current state in the field of anomaly detection. Various approaches to anomalies detection in different type of data and user behavior profiles are described.

The second chapter is devoted to the theoretical side of research. A formal description of the base algorithm is considered, metrics are described that are used for classification, and total complexity of the algorithm is reviewed, as well as its total advantages and disadvantages. As a method of increasing the efficiency of the base algorithm, a formal description of the Personal Profile Adaptive Behavior and dynamic threshold of the anomaly level is considered.

The third chapter deals with the developed experimental system. The main restrictions on the process of calculating the level of abnormality of each user request. Under

existing restrictions a requirements specification experimental system is created. Describes the general structure of the established system, mainly used in its implementation approaches and technologies, their advantages and disadvantages. Further, general issues of influence of the effectiveness of the research environment on the quality of the results are discussed.

The fourth chapter deals with the theoretical ability to evaluate the effectiveness of behavior profiles using different approaches. A common approach to evaluating the effectiveness of profiles is considered that is based on probabilistic characteristics and methods of information theory. In conclusion, a variant of a possible alternative criterion evaluates to "*usefulness of information*" to the recipient is discussed.

The fifth chapter describes all the experiments done. In pursuit of the main target of research, each of the sets of experiments describes the specifics of the behavior profiles from different points of view. Stationary metrics values in conditions of the continuous "normal" behavior are ranked. Experiments are made to assess the impact of characteristics of the data used to train the profile on the accuracy of classification. The dynamics of changes in the quality of classification results in a gradual change in the characteristics of the data being analyzed. The dependence of metrics values on the level of abnormality in the training set is analyzed.

Conclusions include assessment of obtained results and total analysis of the conducted research. In addition, possibilities for the application of used approach to solving other tasks, requiring formalization of personal behavior, are considered.

1. ANOMALY DETECTION IN ELECTRONIC INFORMATION SYSTEMS

The objective of this chapter is to research existent methods of user behavior models. At first, the values of used base terminology are determined. Then we consider the possibilities on detection and counteraction to intrusions, providing by modern Intrusion Detection Systems (IDS). Further one of possible methods of counteraction to intrusions – detection of anomalies is considered in detail, examples of this method are shown, using the methods of signal processing theory, Markov chains of different architecture, ontologies [57, 80] and mobile agents. Besides, we consider one complex solution that unites some different methods at different stages of analysis [82, 79].

At the end, the most frequently used methods of the construction of user behavior profile was considered. The following methods Bayesian networks, ontology engineering and mobile agents was overviewed.

1.1. Basic terms and concepts used

This chapter describes specific terms and concepts used in this thesis. If within the framework of certain chapter specific terms are also used, they will be described in the appropriate section of this chapter.

- *Sensitive data* – in a general sense there are data about a person, which publication can involve his civil rights. Such data are person`s nationality, his political views, psychical features, detail of private life, and history of criminal cases in the past [93].
- *IDS* – Intrusion Detection System. The basic method to detect computer and network intrusions and safety measures and safety providing complex [48].
- *Typical behavior* – the values of the researched object characteristics that can be considered as its average statistical indexes [58].
- *Anomalous behavior* – atypical values of all or some characteristics of the object [58].
- *Detection of anomalous activity* – a process of detection of atypical characteristics and analyzed data [15].
- *Transaction* – single action of user at application layer. In fact, at lower layers one transaction can be presented by plenty of performed actions [33].

- *System work session* – a vector of identifiers of transactions inquired by the user from his initial access to the system (log in) to log out of the system (logout).
- *User role in the system* – an identifier that unites users having similar tasks during work with the system.
- *User class* – group of users by same role.
- *User Behavior Profile* (UBP) – formal data structure allowing to present user behavior [61] as an object (mathematical, statistical, programmatic).
- *Behavior Profile of a Class of users* − General User Behavior Profile trained from data about behavior of plenty of users unified by common role in the system.
- *Personal Adaptive Profile* of *User Behavior* is User Behavior Profile trained from data about behavior of one appropriate user of the target system.
- *OSI* − open systems interconnection basic reference model.

1.2. Concept of anomaly activity

The introduced concept *"anomalous activity"* requires description that is more detailed. Within the framework of this research, this term means behavior that differs from normal user behavior. In a general sense "*anomaly*" is deviation from a norm, some error, difference from typical regularity (target for the system). For detection of anomalies, at first we should to choose a set of target object parameters used for the calculation of its characteristic values. Then, having a set of measured characteristics, it is possible to estimate target object based on their values depending on a set of selected characteristics, as well as on how they describe target object. As a result, the same object can be "*normal*" or "*anomalous*", using the same method, but with different base characteristics.

In this case, term "*normality*" describes any subset of values of all possible characteristics of the system, when deviation from its typical function does not appear.

Intrusion detection system has its own meaning of the concept *"anomaly"*. It includes the current network traffic anomalies, user anomalous data and user anomalous behavior within the framework of work with operating system. General approach in this case includes selecting a set of templates of "*normal*" situation and continuous monitoring of the system at all protected levels.

1.3. Research position related to Open Systems Interconnection model

In this work, detection of anomalous activity at OSI ***application layer*** is used, because lower layers practically do not have differences in solving the task how to detect exactly **anomalous activity** of user.

The OSI model is a basis standard model of open systems interconnection. In general, it is used for description of data flows at different levels of abstraction. Any protocol of model has to interconnect with similar level protocols or with higher-level protocols and/or lower-level protocols. Any protocol of model can implement only similar level functions and cannot implement other level functions. General structure of OSI model levels is shown on Table 1.1.

Table 1.1

OSI levels

Data type	***Layer***	***Functions***
Data	Application	Access to network services
Flow	Presentation	Data presentation and coding
Sessions	Session	Connection session management
Segments	Transport	Direct connection between end−points and reliability
Packages / Datagrams	Network	Determination of route and logical addressing
Frames	Data link	Physical addressing
Bits	Physical	Work with transmission environment, signals and binary data

At monitoring of internal attacks, the following basic analyzed attributes are used:

- data entered by the user, his actions, call functions, queries, API method calls, database inquiries;
- compliance of entered data with required formats, unicity of identifiers, limit minimum and maximum values, for example, payment values or number of working hours.

Exactly at *application layer* there are possibilities for successful construction of UBP and correlation with each action of user. Typical position of IDS in information systems is shown on Fig. 1.1. Data are not encoded and are available for analysis.

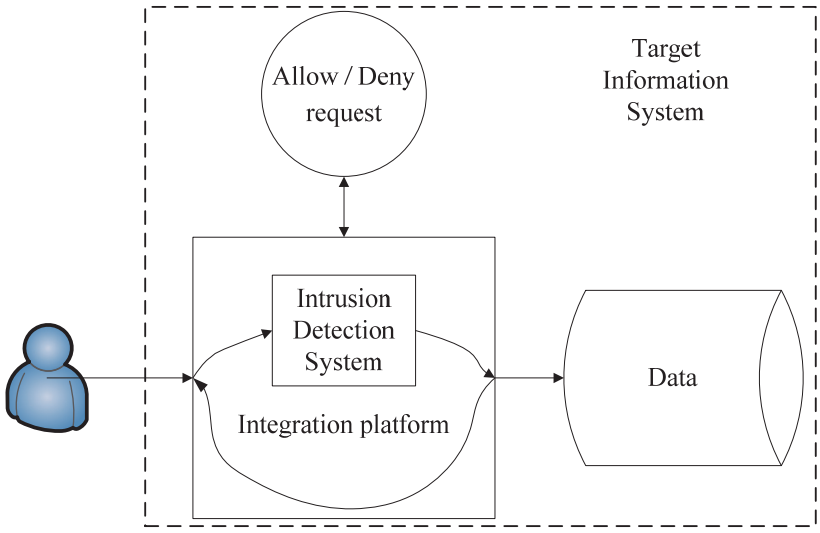


Fig. 1.1 Position of IDS in the information system

The system provides them on the entrance of anomaly level calculation module. Based on user behavior within the framework and terms specific for the target information system it is possible to renew fully cooperation session with the user. Since the similar systems calculate the value of anomalyty level of every transaction of user in real time, the possibility of full renewal of session is used for automatic analysis of new types of attacks and reconstruction of model.

1.4. Intrusion Detection Systems

At present, the basic protection frame of complex information systems from intrusions are Intrusion Detection Systems. They are complex programmatic decisions, providing defense at different OSI layers. Monitoring of these parts of the system is performed:

- system and network activity of user;
- analysis of processes to access the files;
- analysis of access the system processes and libraries;
- monitoring of application processes and users security policy.

Within the framework of providing information safety, the concept *"intrusion"* is a series of actions made by a computer or computer network in order to get illegitimate access

to the stored information [9]. Intrusion can be made from inside or outside of the target system. Basic types of data threats in case of successful intrusion are shown on Fig. 1.2.

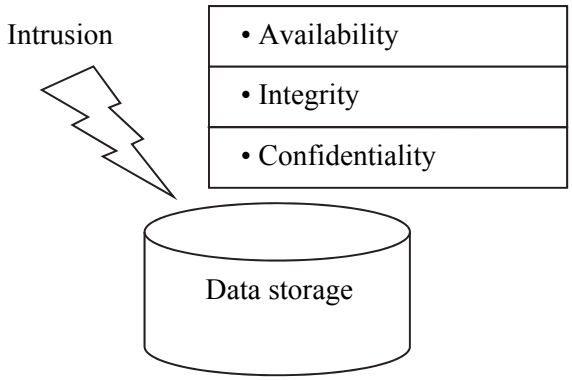


Fig. 1.2 The main types of possible threats to data

1.4.1. Types of user data threats

In most cases the result of a successful intrusion is manipulation of data stored in the system. There are three basic types of data threats (Fig. 1.2):

- threat to availability;
- threat to integrity;
- threat to confidentiality.

The system is considered compromised in case of arising significant probability even one of mentioned threats. In this case IDS basic task is to perform different sets of actions, what can attempt to compromise resource *availability*, *integrity* or *confidentiality*.

Disclosure threat (confidentiality) – arises in case if confidential information becomes available to the third persons.

Integrity threat (integrity) – arises in case of intentional modification of information.

Service refuses threat (availability) – impossibility for legitimate user to get the required information. Depending on non−availability time of target documents, they can to save and lose the status of importance.

1.4.2. Types of the Intrusion Detection Systems basing

Different data sources determine the types of intrusion that can be detected (Fig. 1.3). Each possible type of structure most effectively functions exactly in intended terms.

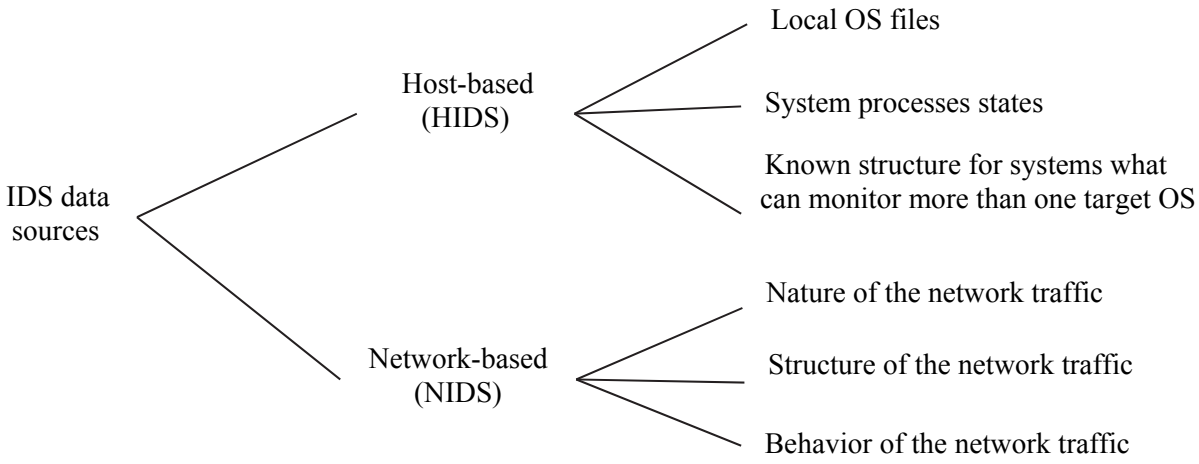


Fig. 1.3 Basic types of Intrusion Detection Systems

- ***Host-based IDS*** – such systems are installed on local computer of user. They allow to execute monitoring of basic activity of computer, such as access to system files, processes, databases, etc.;
- ***Network-based IDS*** – such systems are concentrated on the analysis of network activity. Basic feature of attack is unusual amount and type of network packages or their sequences.

1.4.3. Intrusion detection techniques

Intrusion Detection Systems of any type can use one or more intrusion detection techniques. There are [9] three basic types of intrusion detection techniques (Fig. 1.4):

- Misuse Detection (MD);
- Anomaly Detection (AD);
- Hybrid Intrusion Detection Systems (HIDS).

Misuse Detection based systems use the databases of templates of typical attacks they can detect. Anomaly Detection techniques theoretically allow one to detect new types of intrusion, based on the models used. Hybrid systems try to unite advantages of both approaches.

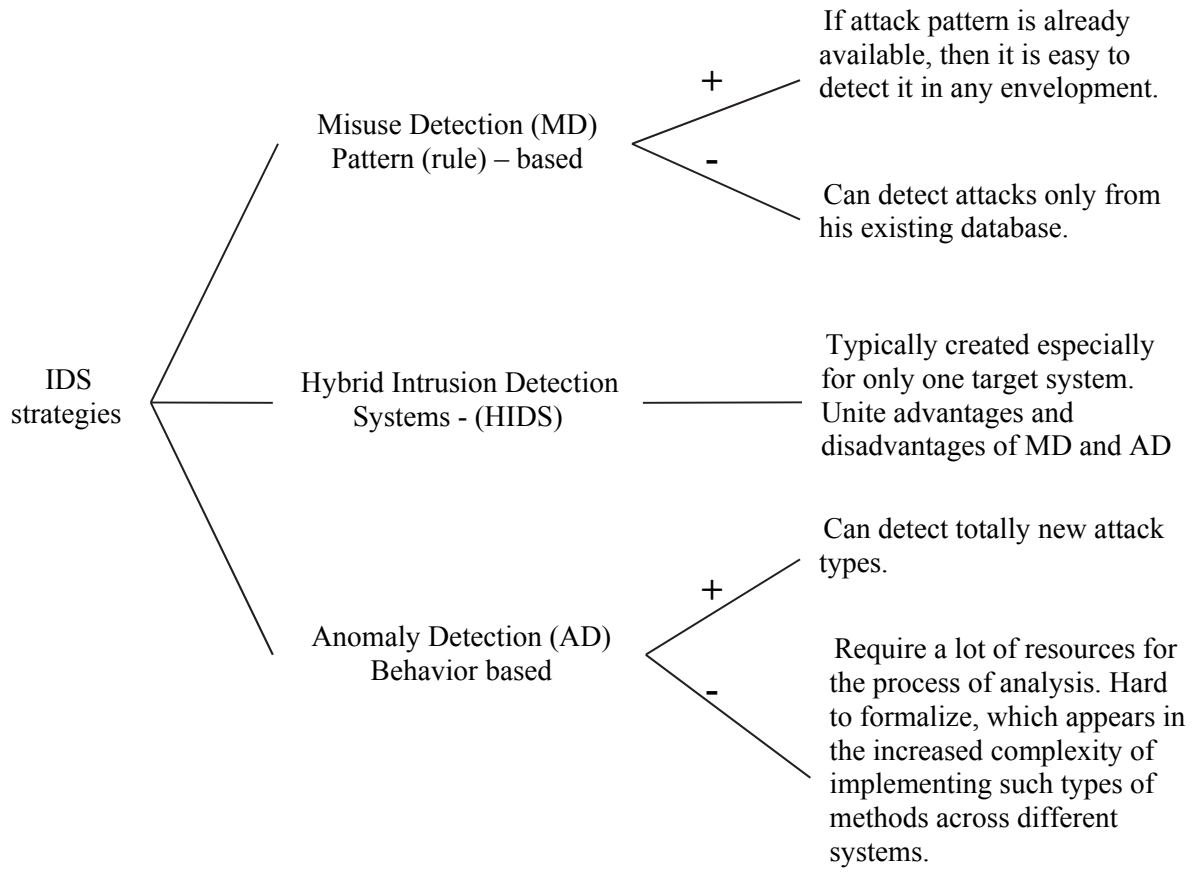


Fig. 1.4 Types of intrusion detection strategies

Hybrid methods of IDS realization can include advantages and lacks of Misuse and Anomaly based methods. Mostly systems, realizing such ideology can find out typical types of attacks, as well as new, not known.

Nowadays, the most popular direction of researches is anomaly-based network intrusion detection because it can detect known as well as unknown attacks. In turn, Misuse Detection is most often used in the environment of real applications as most simple technique. In addition, the user has to understand such important fact that he has to pay for updating intrusion databases in order to prevent new intrusions. Using Anomaly Detection approach, it is not so simple to create commercial basis for constant payments, as theoretically the system has to detect even new types of intrusions in automatic mode.

Basic methods used for application of different approaches for construction of IDS are shown on Fig. 1.5.

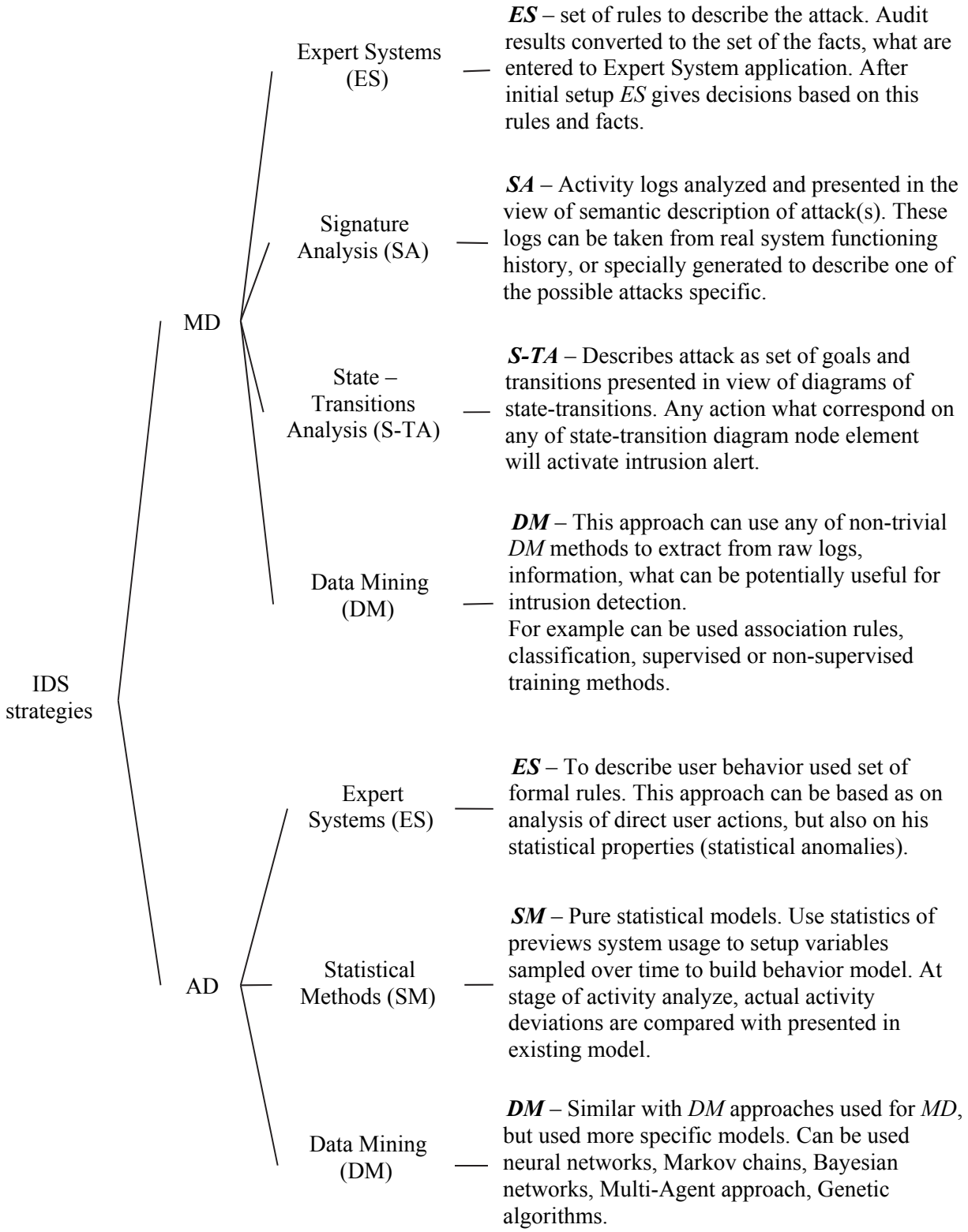


Fig. 1.5 Major approaches to intrusion detection

1.5. Review of existent researches in the field of user anomalous behavior detection

Several methods of anomalous activity detection are considered in this chapter. Most methods used for construction of such systems allow estimating the degree of user behavior anomaly.

At present, there are many methods of construction of model and estimation of its efficiency:

- good results are obtained due to usage of signal processing methods in the task to detect anomalies of behavior;
- Markov chains of different configurations;
- Ontology-based;
- mobile agents.

Each method is considered in this chapter in detail.

1.5.1. Usage of signal processing methods for the analysis of user behavior

In the thesis [49] approach is described, when in the task of anomaly detection in user behavior typical behavior is interpreted as noise, but anomalous as a signal (see Fig. 1.6).

To measure a signal (of anomalous behavior) filtration algorithms are used. If level of signal is higher than some limit, the appropriate action is considered as anomalous.

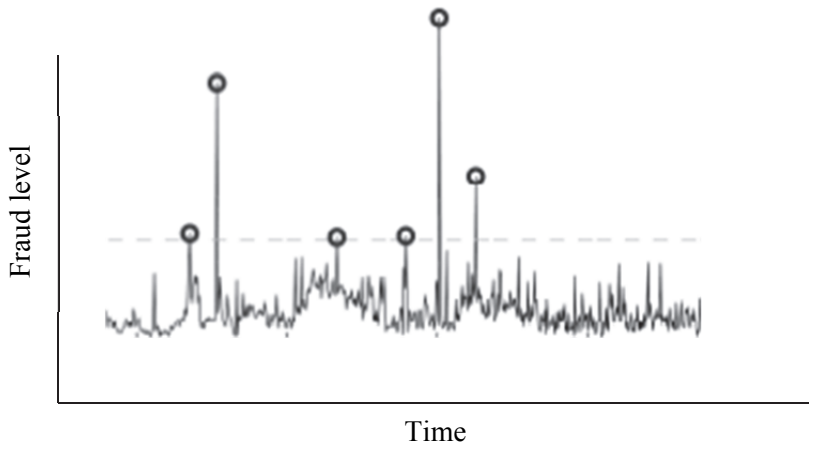


Fig. 1.6 User behavior presentation as a signal

This approach allows decreasing the amount of IDS false trigger, because "*normal*" behavior is filtered and not used for estimation.

1.5.2. Usage of Markov chains

The algorithm considered in [49] shows the model of user behavior as Markov chain. Session of user – system cooperation, consisting of transactions (steps), is analyzed not fully, but only partly – sliding window with ω symbols (user`s transactions in the system). It is compared with already existent base of normal behavior base. For each transaction of user, Hamming minimum distance is calculated for all steps of database.

The model of user behavior is presented as Markov chain $M = (S, p)$, for each step we calculate metrics value $\alpha \in [0,1]$ that means the presence (or absence) of anomalies in behavior. In this case stochastic model of process is shown as (H, A), where H it is ω previous steps history vector, A – value of anomaly. Therefore, filtration is deletion A from H.

1.5.3. Hierarchical hidden Markov models in the task of user behavior analysis

Hidden Markov model (HMM) is Markov model with a set of unknown parameters [30]. In this case it is necessary to define the values of unknown parameters (probabilities of transitions), based on known variables. Similar structures are often used in tasks of computer training, for example, to recognize patterns.

Hierarchical hidden Markov model (HHMM) is expanded idea of Hidden Markov model to present models having hierarchical structure. HHMM is structured multi−level stochastic process. HHMM is used for recognition handwritten input [47], visual recognition of actions [30].

For more formal determination of HHMM the following terms are used:

Σ – final set of states;

Σ^* – various combinations of Σ;

$q_i^d \ (d \in \{1,..,D\})$ – state with index i at the level d;

$|q_i^d|$ – amount of subsidiary states, it is possible to write q^d.

In HHMM a transition between the states at one level is called *horizontal transaction*, between different – *vertical transaction*. Additional variables used to describe it:

$A^{q^d} = (a_{ij}^{q^d}) : a_{ij}^{q^d} = P(q_j^{d+1} \mid q_i^{d+1})$ – probability of horizontal transaction from state i in j for subset of nodes q^d;

$\prod^{q^d} = \{\pi^{q^d}(q_i^{d+1})\} = \{P(q_i^{d+1} \mid q^d)\}$ – vector of initial probabilities for subset q^d;

$B^{q^D} = \{b^{q^D}(k)\} : b^{q^D}(k) = P(\sigma_k \mid q^D)$ – probability that state q^D will give symbol $\sigma_k \in \Sigma$.

Finally, HHNN can be described as follows:

$$\lambda = \{\lambda^{q^d}\}_{d\in\{1,..,D\}} = \{\{A^{q^d}\}_{d\in\{1,..,D-1\}}, \{\prod^{q^d}\}_{d\in\{1,..,D\}}, \{B^{q^D}\}\}.$$

At each level (except the root level) there is the final state, after which a process passes to paternal state for this subset. This condition allows using recursive algorithms for HHMM.

An example of HHMM topology is shown on Fig. 1.7. Detection of anomalies can be interpreted as a task of hierarchical character. Used programs are divided into types of their functionality; programs use function of calls of system from different levels.

Basic part of algorithms, using Hidden Markov Model, is calculation of unknown characteristics of model. For this purpose Baum – Welch algorithm [11] is used.

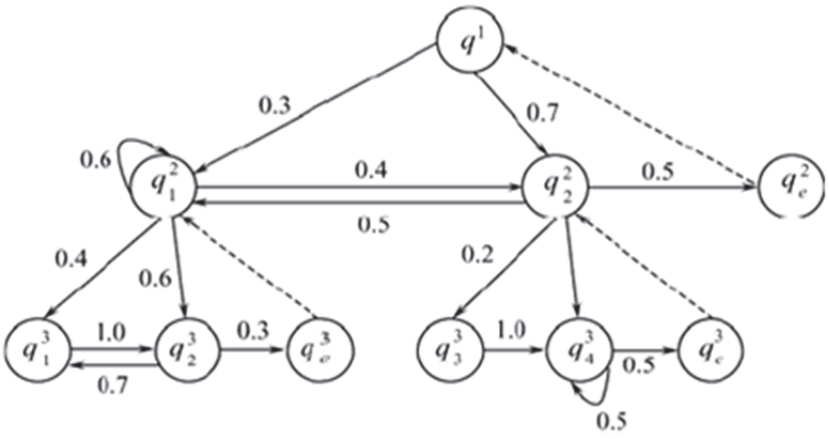


Fig. 1.7 Topology of simple HHMM

In [17] it is shown that as a result of limitations laid on HHMM, time complication for the calculation of anomalies on such network is equal to $O(NT^3)$, where N – the number of states the number of transactions at every step. At the same time a similar algorithm on HMM has complication of $O(N^2T)$.

As an example of this method usage according to the task of intrusion detection in [6] we consider usage of three−level HHMM on statistics of real data of UNIX system calls – New Mexico University (UNM) server.

At training stage, using Baum – Welch algorithm, values of hidden parameters of model are calculated. Then, the base of system calls is constructed for the analysis. At the third stage, testing is executed, sliding window − an analyzer passes on test data, which returns the sequences of data and calculates for each sequence similarity with test sequence O. Probability of anomaly $P(O \mid \lambda)$ is calculated depending on desired exactness.

Based on the same test data, a classifier has been developed that uses simple HMM, and exactness of analysis has been compared with the results of HHMM. In order to compare numeral data, general description of receiver operating characteristic (ROC) has been calculated.

On Fig. 1.8 you can see the charts of anomalous activity determination exactness for both methods. The closer result to upper left corner of the graph is, the more precise it is. Therefore, we can conclude that the method, using HHMM, showed the best results.

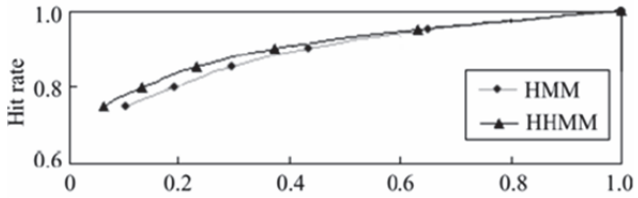


Fig. 1.8 Comparison of accuracy of HMM and HHMM

1.5.4. Ontology in the tasks of anomalies detection

Most attempts of classification of types of intrusions create as a result taxonomy of the attacks that are divided by features. It is difficult to use obtained taxonomies in other systems that differ from the system, for which they have been developed. This limitation cannot be taken into consideration, using taxonomies for storage information about cooperation of elements.

To prevent the same type developments on classification of methods and features of intrusions we decide to use *ontology* [57], as more flexible instrument for their description. Creation of intrusion features ontology will allow to use it in different programs (division of IDS logic and data model) of computer cooperation in automatic mode, the programs will be able to operate by the terms of this area without special installation and help of experts. In addition, it allows creating distributed IDS, when central ontology and queries are used.

In [104] the attempt to create similar ontology and testing of its efficiency is described. Ontology is created based on previous researches, whose purpose is to classify

features of intrusions. In addition, DARPA Agent Markup Language is used [21] and instruments for work with developed model DAML−JessKB [55]. Basic attributes of created ontology are shown on Fig. 1.9.

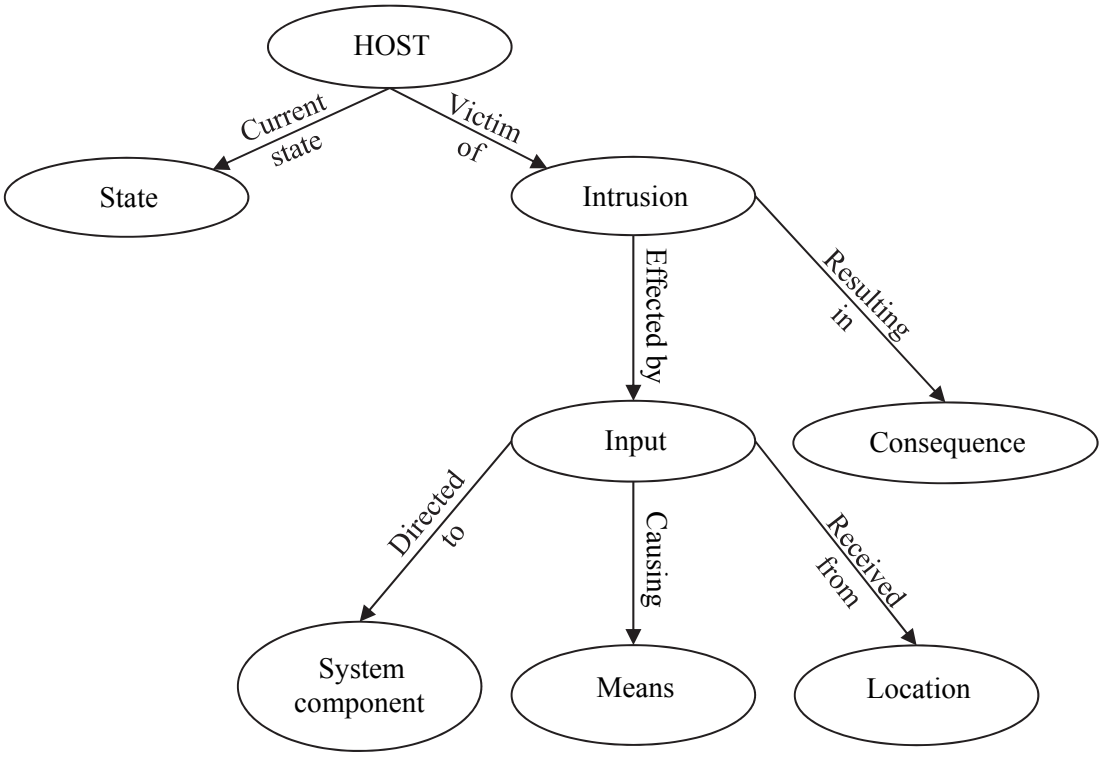


Fig. 1.9 Base research ontology

At creation of considered ontology, about 4000 different types of attacks have been analyzed. As well as existent researches have been considered, and some are included as parts of final ontology. The main categories of model are the following:

- **system components** (most often attacked) − include a stack of network protocols, operating system and applications;
- **essence of attacks** − consists of entered information validation errors, buffer overflow, data limit values processing errors and similar input of unexpected information;
- **consequences of attack** − as a result of attack service refuse, unauthorized access, loss of data confidentiality;
- **source of attack** − attacks are divided into external, local and *external / local*.

Lower levels of abstraction contain more detailed description of its components. For example, "*Deny of Service*" class has subsidiary nodes Syn Floods, Mailstorms, Pings of Death, and all other basic types of "*Service refuse*" type attacks.

As an example famous "*Mitnick attack*" [26] is used. It consists of few attacks at different levels and cannot be fully perceived by typical IDS. However, using IDS central ontology that executes monitoring of different levels, the sequence of error reports can be united in one specialized rule for this type of attacks.

1.5.5. Agent method of intrusion detection

In [22] intrusion detection method is described based on usage of agents. General structure of method is shown on Fig. 1.10.

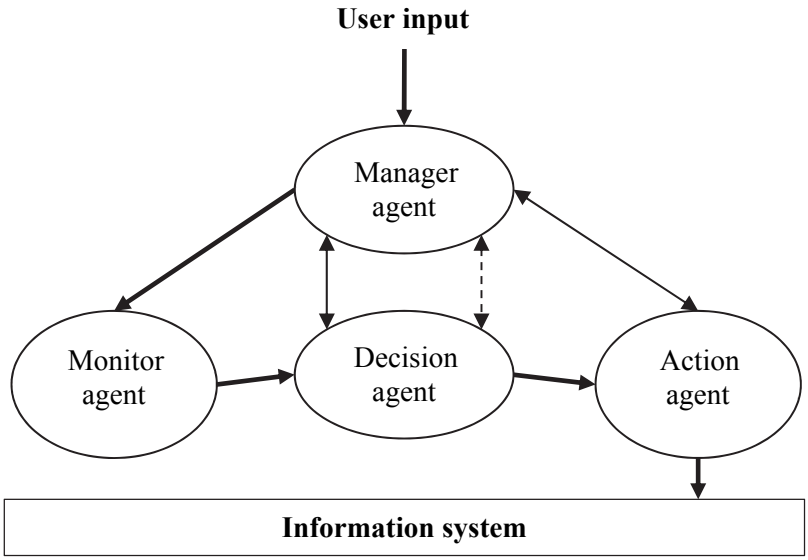


Fig. 1.10 General structure of agent cooperation

The considered system is developed on the basis of tools for multi−agent environments: COUGAAR [106]. Usage of COUGAAR allowed to the authors to be concentrated on basic logic of the system, facilitating technical realization of agents and protocols of their cooperation.

The system is based on cooperation of four agents located at different layers of the system and executing monitoring of events.

- *Manager agent* is coordinator of actions of other agents. Basic task is management of tasks and data flows between other agents. In addition, in case of usage of distributed environment, agent manager communicates with agent managers in other nodes.

- *Decision agent* is responsible for decision making about anomaly level of current analyzed activity. Contains different modules of analysis, such as Fuzzy Logic Module, classifiers, database. Fuzzy Logic Module is used because difference between "normal" and "suspicious" behavior does not have precise borders, and methods to work with fuzzy logic allow decreasing significantly the amount of false trigger.
- *Action agent* reports about status of analyzed system using IDMEF – Intrusion Detection Message Exchange Format. Additionally action agent provides recommendations on possible further actions (for example, to complete a process, to forbid user access to the system, to inform administration).
- *Monitor agent* collects information necessary for the analysis of decisions. Operates at all layers of analyzed system. To increase efficiency this agent can use specialized subject domain.

The following is an example of usage of the system, to detect anomality presence in committed by a user request:

- The user inquires some information; agent manager sends it to monitoring agent for the analysis.
- Analysis agent begins to collect current information from all available layers and to analyze it on presence of norm deviation.
- If there is norm deviation, context will be sent to decision agent.
- Decision agent involves modules of analysis, such as Fuzzy Logic Module, other classifiers to detect context of anomaly level.
- The result of analysis will be sent to action agent that sends the conclusion to agent manager in IDMEF object format.

This system has been successfully tested on few types of test attacks and showed good results, because it has detected 100% intrusions in some test cases. Since 100% on test data does not means even close result on real life, new experiments are pending.

1.5.6. Combined use of different methods

In addition to above mentioned methods, there is interesting example of joint usage of different technologies of data intellectual processing [46] in IDS area.

Within the framework of creation, own IDS (OntoIDPSMA−Ontological Intrusion Detection System and Prevention Multi-agent system) Colombia University developers decided to use different technologies at different stages of current situation analysis.

General structure of OntoIDPSMA is shown on Fig. 1.11. Each input TCP package passes a few stages of analysis using different technologies, and at the end IDS provides a conclusion about possibility of its assess to internal target information system.

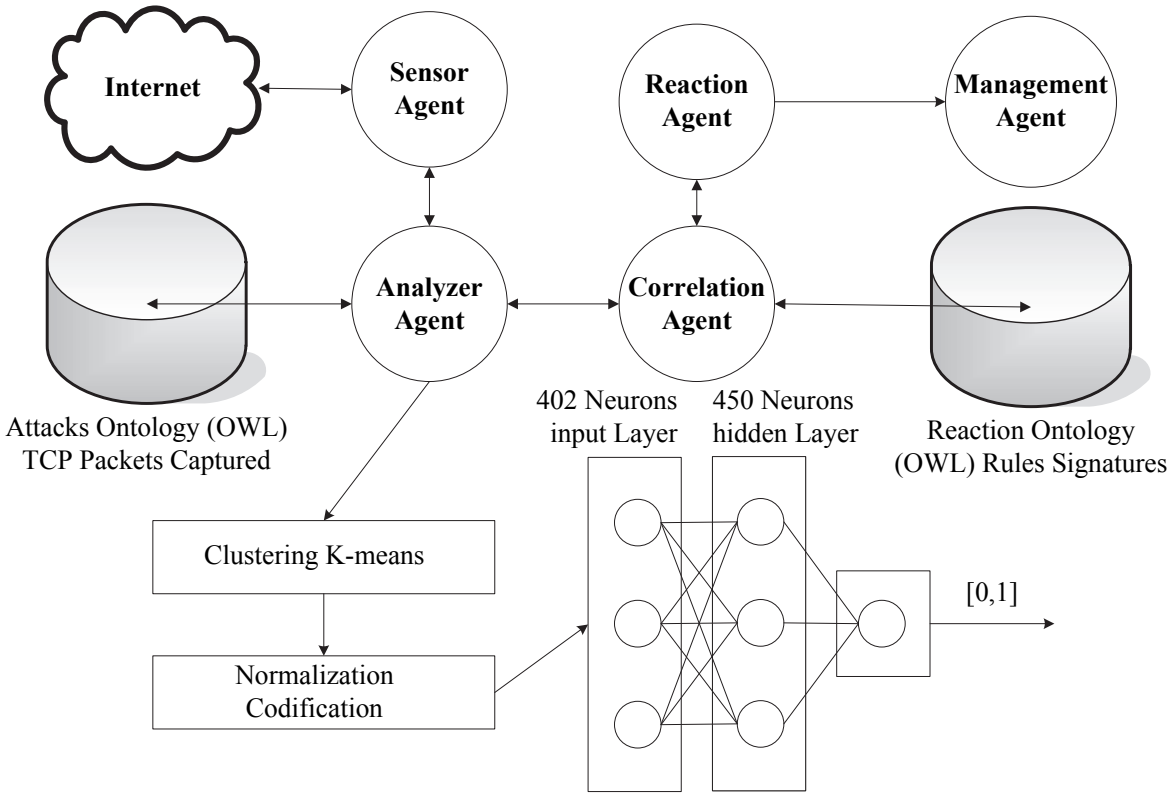


Fig. 1.11 General structure of cooperation of modules OntoIDPSMA

Basic part of analysis is made using agent method that is similar to above mentioned. Agents are shared with information in IDMEF format. To provide knowledge about features of attacks and recommended corrective actions ontology is used. Agent – ontology communication is executed using IDS language (Web Ontology Language [2]). Types of input data (Internet Protocol, inquired ports, data, types of protocols, etc.) are classified using *K−means* method; result of classification by all parameters is normalized and given to neural network that finally provides the result about the presence of anomaly in the analyzed query.

Authors consider that due to usage of complex method of similar structure, efficiency of final system is higher than efficiency of usual IDS.

On the other hand, obvious complexity of such system will not allow unifying it for the use in the systems that differ from the described. A good result is explained by very deep integration with the target system that will not allow getting the same level of anomalies detection within the framework of the system with different structure.

1.5.7. Advantages and disadvantages of the approaches reviewed

In this section, different approaches aimed to solve the problem of detection of anomalous behavior of the user are reviewed. Each of them has its advantages and disadvantages. On that basis, in section "Requirements to system" criteria were obtained for assessing the suitability of the considered approaches. Results are listed in Table 1.2.

Table 1.2

The presence of the desired characteristics in the systems examined

	Self−learning	***Input data minimum***	***Maximal isolation***	***Processing speed***
Filtering Approach (section 1.5.1)	-	N/a	N/a	+
Markov Chains and Hemming Distance (section 1.5.2)	-	+	+	-
HHMM (section 1.5.3)	+	-	-	-
Ontology−based approach (section 1.5.4)	-	-	+	-
Agent−based approach (section 1.5.5)	+/-	-	+	-
Complex solution (section 1.5.6)	+/-	+	+	-

It is evident that none of the reviewed approaches provides a complete implementation of all the existing requirements. As result, appeared requirement to create new approach what will cover all existing requirements and try to unite advantages from methods what have been reviewed in this chapter.

1.6. Approaches to construction of user behavior profile

One of the most perspective methods of detection of user anomalous behavior is construction of user typical behavior profile and further calculation of difference between user current behavior and created profile [77]. In this chapter, different methods in this area are described.

1.6.1. Using of Bayesian networks

Bayesian network [83] — is graphic probabilistic model represented by various variables and their probabilistic dependences presented as set of of random variables and their conditional dependencies via a directed acyclic graph. For example, Bayesian network can be used for the calculation of probability patient`s disease based on presence or absence some symptoms, based on data about dependence between symptoms and diseases.

In [56] the method of detection internal threats is considered within the framework of information system. This task can be solved by construction of user behavior model based on *hybrid Bayesian networks* (multi-entity Bayesian networks (MEBNs)). Typical method is used, when user model based on his previous behavior is compared with his current actions, but significant difference between indexes means presence of anomalous behavior. As proof of idea method, test realization is described.

If the user knows about the type of security system used in protected system, he can very slowly change his behavior, that does not show significant difference from session to session, and only usage of long history of usage of the system can find out the fact of continuous change of his behavior. It does not mean that he makes something bad maliciously, but it is a reason for security system to pay more attention. In spite of presence specified procedure in person`s behavior, it is not predictable. A lot of external and internal factors impact on behavior, but within the framework of information system the amount of possible actions is limited, that allows constructing final model.

1.6.1.1. Bayesian networks in the tasks of construction user behavior profile

The Bayesian networks that model the sequences of variables are called dynamic Bayesian networks. Bayesian networks, in which discrete variables and continuous variables are present, are called hybrid Bayesian networks. Bayesian network, in which arcs encode not

only relations of conditional independence, but also relations of causality, are called *causal Bayesian networks* [53].

Bayesian probability theory is powerful instrument of construction of models in the terms of uncertainty. Important advantage of these models is possibility to combine expert data, as well as the results of experiments during a long period of time, increasing their accuracy. At present combined method becomes more popular, when the Bayesian models are used with the theory of graphs. It allows constructing more difficult and exact models based on plenty of cross hypotheses. The Bayesian network is probabilistic model as oriented graph to represent quality relations and local distributions of probabilities to show quantitative information about types and levels of concept mutual relations.

On Fig. 1.12 a part of simple model of user behavior is shown at the query of document. The model is shown as totality of casual values within the framework of general hypothesis. For example, casual value *GlobalIntention* shows probability of presence malicious intent in a query. It in turn depends on the value of casual value *Motive*. At absence of motivation, the user probably will not have illegal intentions, but for motivated user this probability increases significantly.

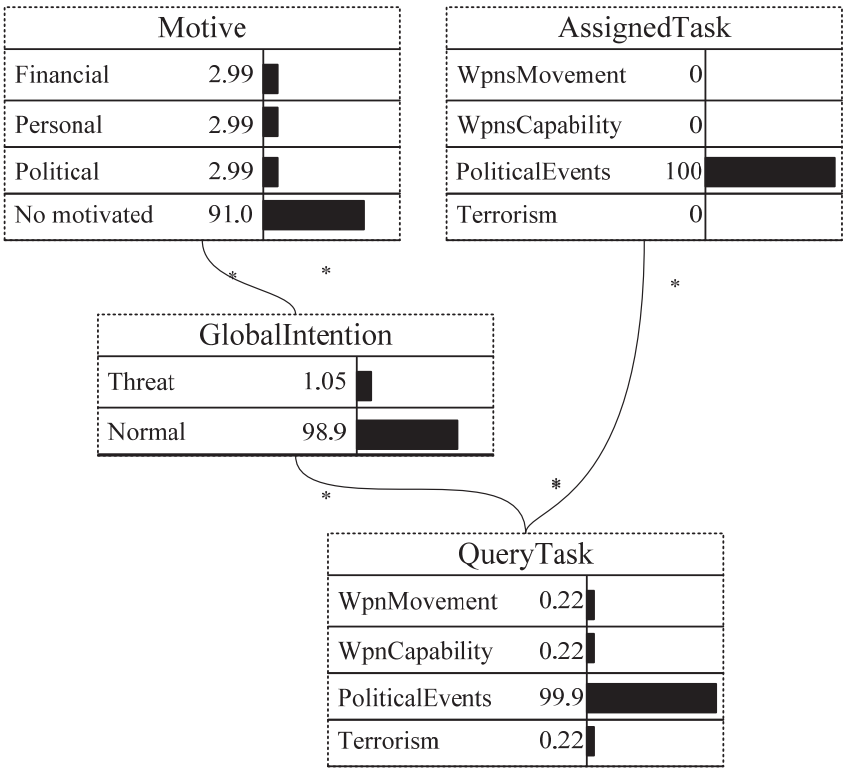


Fig. 1.12 Part of Bayesian model

1.6.1.2. Hybrid Bayesian networks

Standard Bayesian networks are limited by problems, in which one set of random variables is used for all problems, and only features differ depending on the problem. Presentation that is more flexible can be obtained, using as nodes only untypical situations.

For this purpose hybrid Bayesian networks are used, which operate with parametric structures called MEBN Fragments (MFrags). Each MFrag is module component that represents little enough, all-sufficient and conceptually meaningful part in support or refutation of current hypothesis. Using MFrag it is possible to model different hypotheses, creating the chains of associate concepts. Their modular structure allows to combine flexibly and to use repeatedly before created elements in new researches. It is possible to say that each MFrag encapsulates the functions of element of simple Bayesian model.

For example, on Fig. 1.13 it is shown how by MFrag to present the model. Each MFrag contains a set of casual values (marked with white), and their parameters of distribution are also presented in models, input casual values (marked with light grey), their values depend on internal casual values, and context casual values (marked dark grey) that have to equal *True*, in order to show significance of distribution specified in resident random variables MFrag. All casual values got values called as *entities*. For example, query of MFrag shown on Fig. 1.13, when entity u (user) is equal to value *Performing User (q)*, for entity *q* (query) − i.e., when an appropriate user makes a certain query.

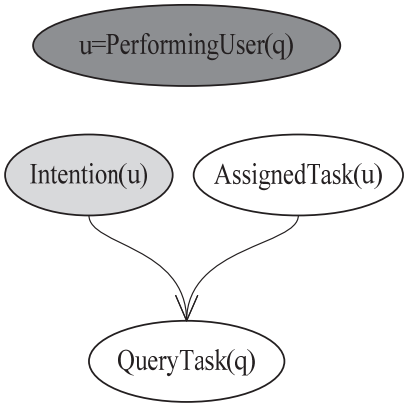


Fig. 1.13 Example of MFrags dependences

Local distributions and their parameters, used in Mfrag, are obtained from real data. Such method can be used for description of difficult models with usage of plenty of participants, documents and computer systems. Final model is constructed as the Bayesian

network that consists of united Mfrag. This model is used in real−time to describe user behavior. It can be used for detection of anomalous behavior, as well as for its research and analysis.

1.6.1.3. Experimental model

To carry out experiments the test system consisting of seven MFrag and modeling user behavior that makes queries to the system and works with documents has been developed.

The following entities have been developed:

- **User-MFrag** describes one user. Describes identification data of user and his reasons and intentions.
- **User Background** − describes three possible reasons to do harm to the system: political activities, personal background, and financial background.
- **User Assignment** − data about geographical region of user and his tasks.
- **User Intention** − current classification divides users into "*usual*" and "*dangerous*". Obvious that during one session the user can change his state. In addition, users can have general intentions that do not change during many sessions.
- **User Other Intention** – the purpose of created system — not only to create the identifier of malicious user, but also to find out essence of potential threat. Therefore, it is important to take into consideration also other features of user behavior in the system.
- **Document** − documents have sources, regions and value of classifier. In addition, each inquired document has a degree of compliance with a query.
- **Query** − users make queries and receive a response.

1.6.1.4. Simulation results

The purpose of experiment was to learn to distinguish the type of user (normal/ malicious) base on his actions (queries of documents) in a certain period (made during previous sessions). Two identical models have been created.

1. The first model (ground truth) was used for modelling of intentions and user behavior.

2. The second model (inference) was intended for determination of presence malicious intent in current behavior.

Test data about 100 sessions were generated for each user. Each model consists of set of queries of different documents. The behavior of 192 users has been modeled, the type was known at advance – normal or malicious.

The results of normal behavior analysis are shown on Fig. 1.14. We can see that during all 100 sessions, estimation of probability of normal behavior was not below 0.9. This is good example of *normal* user, what have not anomalies in his behavior during whole work session in target system.

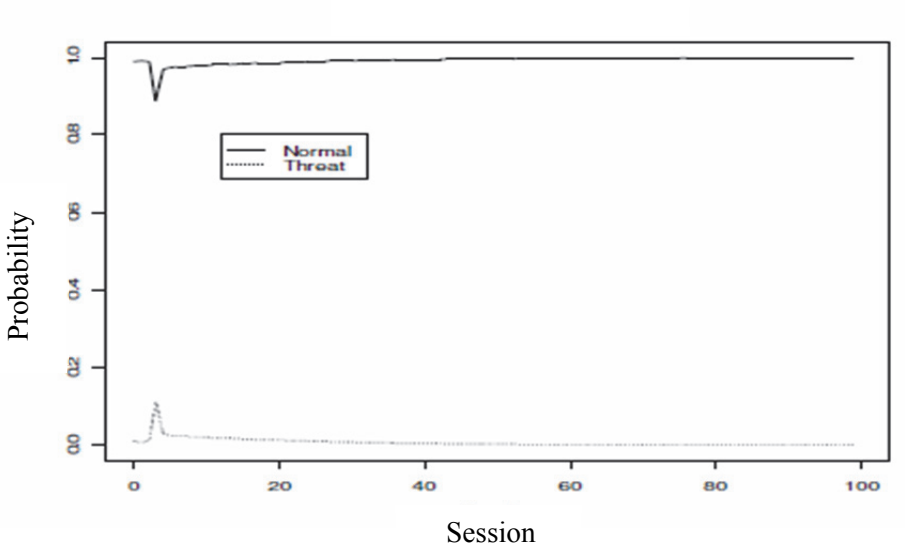


Fig. 1.14 Normal user behavior

On Fig. 1.15 probability of normal behavior is shown in case if the user tried to hide his malicious behavior.

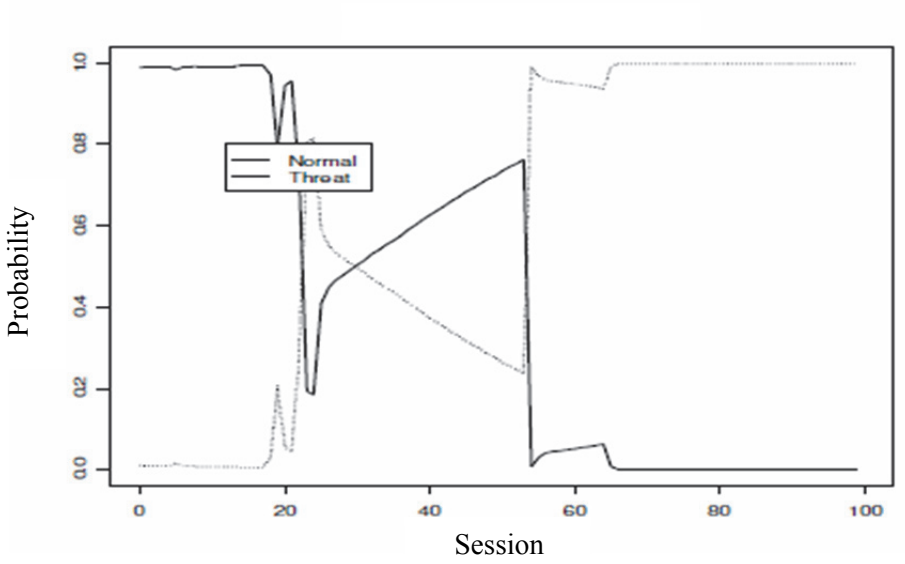


Fig. 1.15 Anomalous behavior after several iterations of analysis

Graph on Fig. 1.16 shows a case, when malicious user has not been detected. It is easy to see that in this case his behavior is estimated as "*normal*" during all sessions in spite of the fact that his actions are illegal.

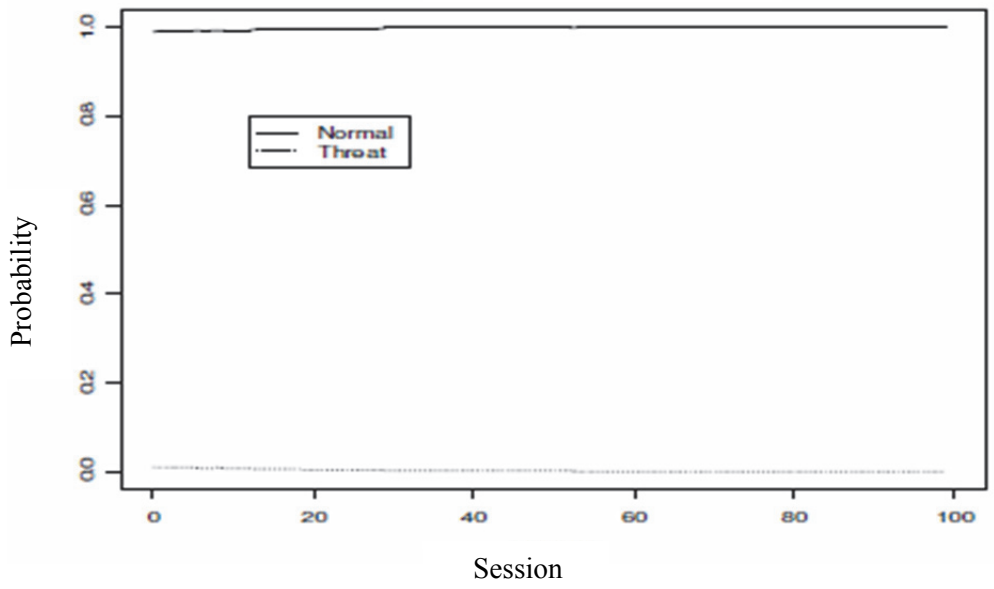


Fig. 1.16 Undetected malicious user

Unlike Fig. 1.15 where a case of postponed detection of anomalous behavior is shown, on Fig. 1.17 typical case of malicious user is shown.

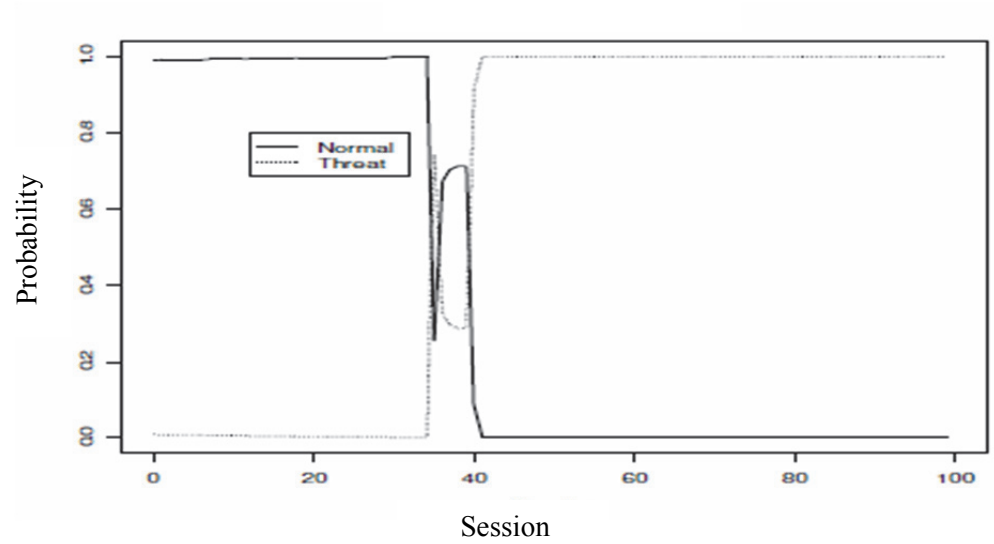


Fig. 1.17 Typical detection of malicious user

Experiments were carried out on the basis of artificially generated data, and the purpose was to prove functionality of method, what was shown. Further, authors plan to use

data of real existent system, to range detected anomalies of behavior, to increase significance of characteristics used in description of user and to use security policy close to reality.

In addition, authors consider that important direction of further researches is usage of Data Mining methods for data preparing and further analysis of obtained Bayesian network. In particular, it is suggested to create general ontology of documents, representing their internal semantic dependences for more precise estimation of compliance of obtained document with user query.

1.6.2. Usage of ontology

French scientist L. Razmerita in his paper [88] describes the method of modeling user behavior, using methods from area of Semantic Web [40] and with usage of ontology [57].

The approach using ontology for modeling becomes more popular due to flexibility, quality of offered methodologies of data and knowledge conceptual management, possibilities on distribution and repeated usage of knowledge.

This approach [1] considers modeling of electronic system user behavior, using ontology in the context of Knowledge Management Systems (KMS) Usage of ontology allows describing the features of subject domain in general; it will allow in future using the obtained description for decision of other tasks in other applications and by other groups of researchers. Additionally, by complication of ontology it is possible to describe in detail behavior of each user.

The process of ontology development is difficult and labor intensive task that requires knowledge in different areas, such as engineering, software development, object-oriented programming, modeling theory, artificial intelligence, etc.

In general, the process of ontology development consists of three simple steps:

- knowledge collection;
- encoding;
- ontology integration.

The developed ontology will be structured in accordance with IMS LIP [45] specification: „The purpose of specification is determination of set of modules that can be used for data import, as well as for data receipt from information server compatible with IMS".

In order to describe the user, each IMS package is structurally divided into 10 information sub-groups. These groups are following: *Identification, Goal, QCL*

(Qualifications, Certifications and Licenses), Accessibility, Activity, Competency, Interest, Affiliation, Security Key and *Relationship*.

Identification describes personal data. *Affiliation* includes data about person connection with target organization. *QCL* includes the list of person skills, his diplomas, certificates and licenses. *Competency* describes other formal and informal skills, as well as person`s employment history. *Activity* describes person activity related to training within the framework of his professional duties. *Accessibility* concepts related to possible specific features and user requirements. *Interest* describes hobby and interests. Concept *Goal* includes basic and secondary purposes of user.

However to describe behavior, these elements are not enough, so additional concepts are introduced. Behavior models characteristics of user – system cooperation. Data for this module are received from log-files of user activity history in the system. For construction of heuristics suitable for rules forming, such concepts as *Type_of_Activity*, *Level_of_Activity* and *Level_of_KnowledgeSharing* have been added.

Type_of_Activity describes type of user behavior in the system, it prefers mainly to read, to write, to remain unnoticed. In addition, each type of behavior can be classified as "*very active*", "*active*", and "*passive or / active*". In addition, the third type of classification is based on estimation of the analysis of user knowledge distribution level; users are divided into *Unaware*, *Aware*, *Interested*, *Trial*, and *Adopters*.

The next step is ontology encoding using one of formal languages. Alternatively, it can be created using ontology editor, such as OntoMat, OI-Modeler, KAON or more often Protégé [68].

Then created ontology is introduced and used within the framework of general created system.

Ontology introduction. At this stage, mechanisms of modeling and personification of model are realized as a set of intellectual services. On Fig. 1.18 architecture of created system Ontology-based User Modeling framework (OntobUMf) is shown.

The data used for user model creation have accurate structure and are partly based on profile provided by the user, as well as on the conclusions of one of category extractors, represented as intellectual service working in automatic mode. The task of category extractor is classification of user type based on user activity type in the system. Any action of user in the system is registered in transaction log.

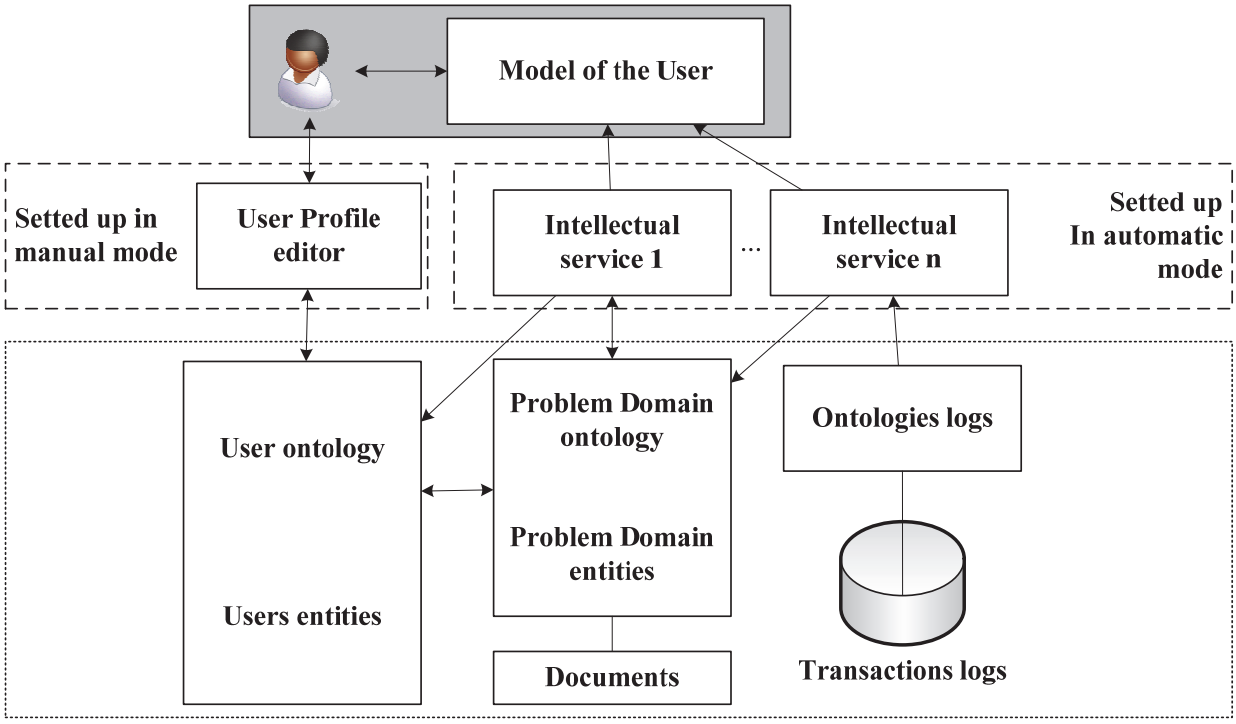


Fig. 1.18 Architecture realized

User profile editor is specialized ontology editor available for end users and for management of personal profile description.

Intellectual services due to module system OntobUMf can use various types of algorithms as intellectual services that determine type (role) of user current behavior.

Common tasks of the service, what should be realized to become possible detect anomalyous activity of user using approach described are the following:

- Updating and support of user model based on data received from *category extractor*.
- Providing of individual analysis for specific types of users.

Adaptation and personalization is a process of the system adjustment for the appropriate user behavior features. For example, adaptive can be graphic interface, data structure and availability, security policy. General task of this process is *providing of necessary information at necessary time to necessary users*.

User interests' ontology created within the framework of this paper can be used for different purposes. Depending on the task, specific intellectual service of *category extractor* is realized that performs classification of users.

1.6.3. Multi-agent method for modeling user behavior

In paper [95] one more possible method for *Grid* information system [31] user behavior modeling is considered. This test system showed good results for introduction in real projects.

Now GRID information systems become more popular in applied environmental science, calculation decision-making. Current methods of security providing in such systems are constructed based on PKI (Public Key Infrastructure) [93]. Such method provides implementation of usual authorization, authentication protocols, delegation of powers and exchange by certificates. However, detection of user anomalous activity by these facilities is not effective. The theme is actual; there are a few attempts to solve it [94, 97] but at present sufficient accuracy of such attacks detection was not achieved.

The analysis of other existent methods of information system user anomalous behavior detection showed that they are ineffective within the framework of specific of *Grid* systems. Therefore, we set the task to develop new model, taking into consideration the features of such systems. For example, if typical work of user is successive implementation of plenty of different commands, within the framework of work with *Grid*-system the amount of executed tasks is less, but each requires more resources.

On Fig. 1.19 general sequence of actions executed using user behavior model for detection his anomalous activity is shown.

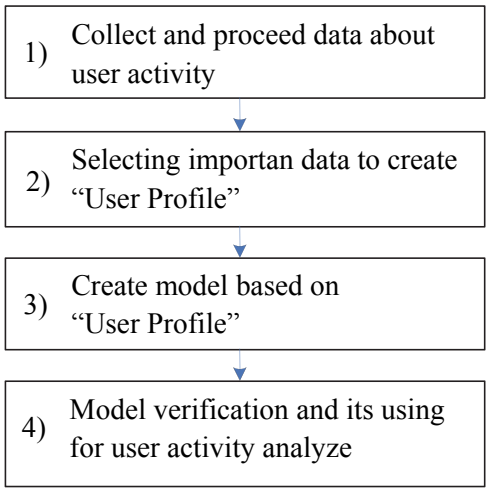


Fig. 1.19 Typical activities at behavior modeling

The largest difference, using different methods, is observed at step 3, where different methods can be used. In the considered paper at this stage it is suggested to use Neural Network Method [42], and for realization of all system – Agent Method.

Neural networks are popular method of classification. Multi-layered feedforward neural network has been used. As input data, the following set of characteristics has been used:

{S, ET, CPU, WT, CW, ES, CT, STD, RAM, VM, VO, RB},

where S (*Site*) − a node on that a task is carried out;

ET (*Execution Target*) − a resource of node, executing a task;

CPU (*CPU Time*) − processor time, required to resource for implementation of task;

WT (*Wall Time*) − total time of implementation of task;

CW (*CPUWall = CPU/W*) – processor time to total time of implementation of task ratio;

ES (*ExitStatus*) − status of task completion (successful or with error);

CT (*Creation Time*) − task dispatch (creation) time of *Grid*-system;

STD (*Start Time Difference*) − difference between task implementation beginning time on selected resource of *Grid*-system and task dispatch time to it;

RAM − volume of used operating memory;

VM − volume of used virtual memory;

VO − identifier of belonging to virtual organization;

RB − identifier of task resources broker.

For each user his personal neural network that classifies his behavior as normal (*1*) or anomalous (*0*) is constructed.

To carry out experiments, data about users work in the system GILDA [32] European project EGEE [27] have been used. In general, a database consists of more than 34 000 records of users' actions access log. For monitoring of all available states of the system, distributed system GridICE that is integrated with local resources monitoring system is used in GILDA.

In the process of programmatic realization of test system, raw data about users' behavior have been regenerated in XML format and then divided into training and test selections in ratio 85% to 15% accordingly. A Neural Network with direct information distribution and one hidden layer has been used, and trained using the method of error back propagation. For the calculation of network structure optimal values A/B testing has been carried out, which showed that the best configuration are 20 neurons of hidden layer, showing 85,81% classification accuracy. As well as values ($\eta = 0.3$, $\mu = 0.15$) have been bound for weighting factors and training parameters.

Final testing is usage of user substitution procedure, when after initial usage of data about user behavior, other (illegal) person data have been dispatched to a classifier. Results showed that legal user classification level is equal to 99.14%, but in case of user substitution, correct classification has been made in 99.30% cases. The results are sufficient for usage in real systems.

General structure of the system using Agent Method. The major feature of systems GRID is their distributed, heterogeneous structure. In addition, it is important to isolate the module of detection anomalous activity from all other system. Thirdly, it is necessary to

provide co−operating with monitoring system for receipt information about the tasks started by users and CA (Certificate Authority).

All these requirements are satisfied by Agent System Paradigm. In this case, to estimate presence of anomaly in the action executed by the user, the agent is used. Agent encapsulates neural network of user model and methods of receipt information required for the analysis.

Monitoring system GridICE is centralized. To collect information about tasks executed by users, Grid Resource Information Service (GRIS) is started on each resource. In turn, domain services Grid Index Information Service (GIIS) co−operate with local services GRIS, aggregate the obtained information and sent it to the centralized server GridICE (Fig. 1.20).

For realization of user behavior model, in the terms of system created, the following types of agents have been realized:

- Agent, encapsulating GRID system user model (User Agent). It is activated after implementation of the action inquired by the user, contains personal neural network of current user model, statistical data about previous behavior and result of analysis of presence of anomaly in the executed transaction.
- Agent controller manages creation and behavior of user models agents. As well as its task is to inform security system about cases of anomalous activity detection.

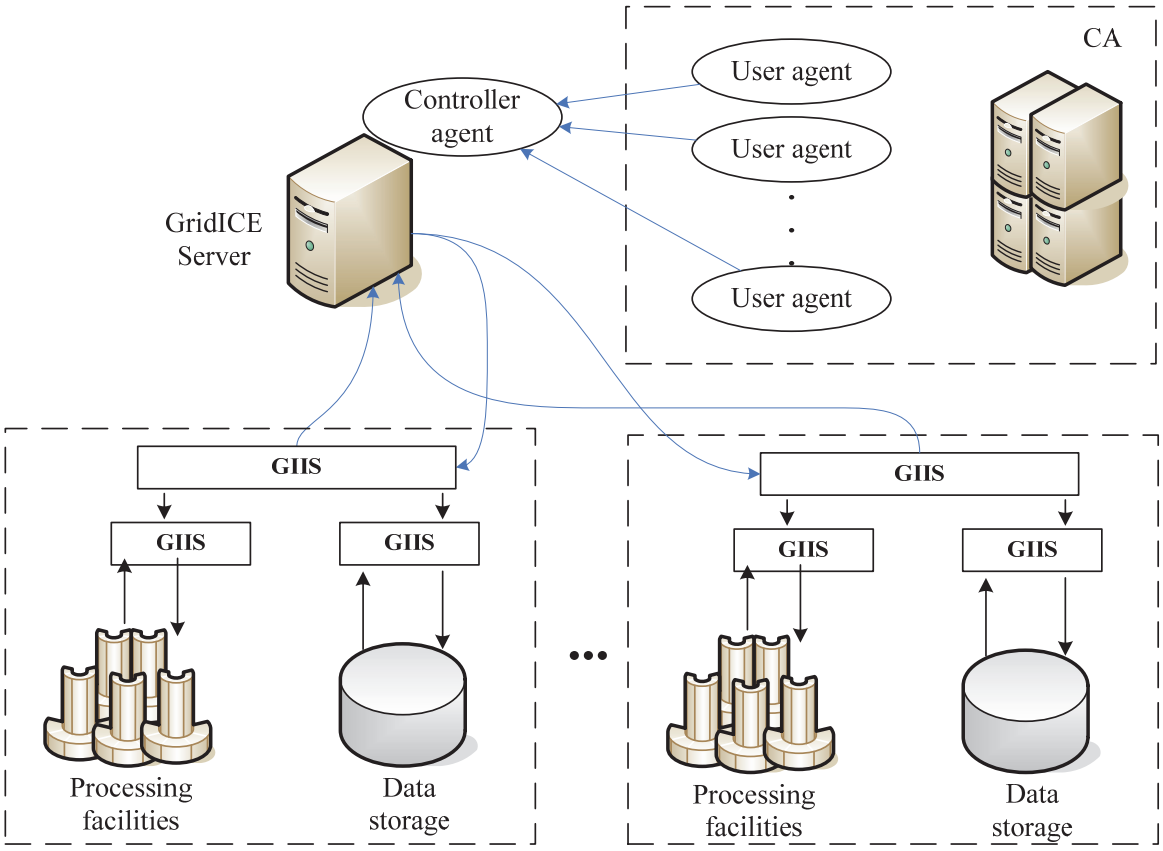


Fig. 1.20 Agent – system cooperation

1.7. Summary

At first, it is important to determine the values of used base terminology. Introduction of detailed description of all terms used in a chapter allowed avoiding ambiguity of their interpretation and to obtain clear presentation of material.

Then intrusion detection and prevention possibilities offered by modern Intrusion Detection Systems are considered. Classification of such Intrusion Detection systems provided forming of integral form of methods of their construction, solved tasks, used approaches. In addition, it allowed determining the most appropriate approach within the framework of the solved task.

One of possible intrusion prevention methods – anomaly detection – is considered in detail, samples of realization this approach are described, using methods based on the signal processing theory, Markov chains of different architecture, ontologies and mobile agents. Complex decisions unifying several different approaches at different stages of analysis are also possible.

In conclusion, most often used approaches for construction of user behavior profile are considered. Methods, using different variants of Bayesian networks, ontology engineering, mobile agents, are considered.

- Research of the methods used for construction of formal user behavior profiles provided possibility for development of own type of behavior profile, taking into account all required functionality.
- The analysis of the approaches used for solving anomaly detection task provided the basis for the choice of the most appropriate variant of realization its own system in this area.

2. MARKOV CHAINS IN THE TASK OF DETECTION ANOMALOUS ACTIVITY

This chapter describes base algorithm applied for construction and use of User Behavior Profile (UBP) [76, 74, 107]. Formal description and analysis of complication of algorithm basic elements is provided. Flow−graphs of basic used operations are described. Impact of internal parameters on quality of classification is considered, as well as general approach for its testing [71]. Basic advantages and lacks of **base algorithm** are described. Approach for improvement of basic algorithm by introduction **P**ersonal **A**daptive **P**rofile of **U**ser **B**ehavior [108, 72] is considered; possible lacks and advantages caused by introduction of the approach are described. Possible method to increase the efficiency of classifier is also considered, introducing dynamic threshold of the anomaly level.

2.1. Used terms

This section describes the basic specific terms used in chapter following. Some of this terms also used in next chapters.

- *The atomic user action (transaction)* – user access to the system service fixed at a certain point in time (data loading, copy, query, etc.).
- *Role of user* − determines a range of tasks and interests of target user in the system. Users united by same role inside of target system will use the system similarly in comparison with representatives of different roles.
- *Alphabet* Σ – full set of all possible transactions in target informational system. In other words, it's all types of actions what user may request.
- *User work session is a sequence* of user queries between user logging in (authorization) and logging out (implementation of the last transaction). Last transaction not always will be *"Log out"*, for example session may be not properly closed and ended in automatic mode by inactivity timeout.
- *Suite* – well ordered in time number of records about user`s transactions (the sequence of alphabet Σ elements), carried out during one session.
- *Trace* – number of suites used for construction of model.

- *Window* – number, amount of alphabet Σ elements; their set forms one node of Markov chain.
- *Transition* – a pair of states (*s, s'*) that determines transition from *s* to *s'*. Every state and transition is related to the meter of transitions amount.
- *Metrics* – in terms of used algorithm, it is numerical estimate of presence of anomaly in the analyzed transaction, which is calculated based on current model of user behavior for any requested transaction.
- *Classifier* – function: $f: \sum * \rightarrow B$, based on current metrics value, returning response 0/1 ie ($B = \{0,1\}$), feature of belonging current action of user to anomalous activity. Thus *one* in this case is a feature of "*bad*" trace-presence of anomaly, and *zero* is a feature that this trace is "*good*" − absence of anomaly. As transition to the next state is determined by only current and previous states (Markov chain has no memory), it is not necessary to analyze complete current way.

2.2. Base algorithm of construction of user behavior profile using Markov chains

As a basic form of data presentation about user behavior, Markov chain – a weighted oriented graph having *Markov property* [35, 50, 62] – is used in the thesis. The sample of simple graph of Markov chain is shown on Fig. 2.1.

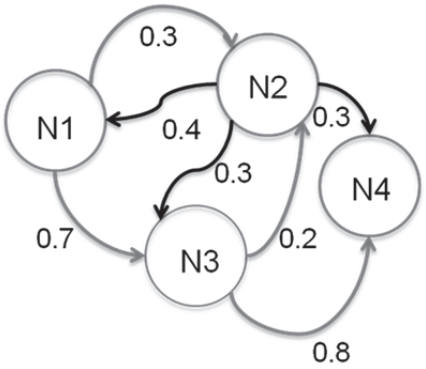


Fig. 2.1 Simple Markov chain

In turn, Markov property – property of absence of memory, when the current state is based only on transition probabilities from the previous state, and detailed history of visiting

the previous states does not impact on the current situation. In addition, limitation of complete probability is imposed on the graph of Markov chain, when the sum of sent probabilities for every node is always equal to 1, i.e.: $\sum_j P_{ij} = 1; \forall i$.

Usage of Markov chains in the task of presentation user behavior profile allows applying large volume of theoretical knowledge, accumulated by mathematical statistics and probability theory in this area. However, it is significant that in the considered method Markov chains are used mostly as comfortable data structure to present user behavior, when limitations imposed on usual oriented graph [39] add Markov property. Exactly Markov property allows not walking around behavior graph at each new query of anomaly level calculation, but to analyze the node describing current behavior and transaction inquired by the user.

Graph describing behavior can have plenty of nodes and arcs, and a necessity to analyze only some subset allows usage of time and resources more effective. Used method can be divided into the following parts: *training*, *usage* and *testing*.

2.2.1. Training of the profile

At training stage, UBP is constructed on the basis of data about usage of target system. Histories of behavior of users united by same role inside of target system are used for creation of general profile of user typical behavior. General structure of used method is shown on Fig. 2.2.

Users, in the process of system using, in usual mode, generate work sessions – as sequences of requested transactions identifiyers. One or more such sessions when can be used for UBP training, describing typical behavior. Training process not proceeds in same time as user uses target system, only complete sessions can be used to educate behavior profiles. In addition some sessions can be excluded from education process if his quality level less than some internal criteria (for example, count of anomalies detected exceeds some threshold defined in system).

UBP is data structure that includes a graph describing the features of behavior and additional metadata (see Fig. 2.3) what may be necessary for description and storage of profile data, for example as metadata stored: target role of profile, different statuses and identifiers, last access timestamp, and other necessary common or target system specific data.

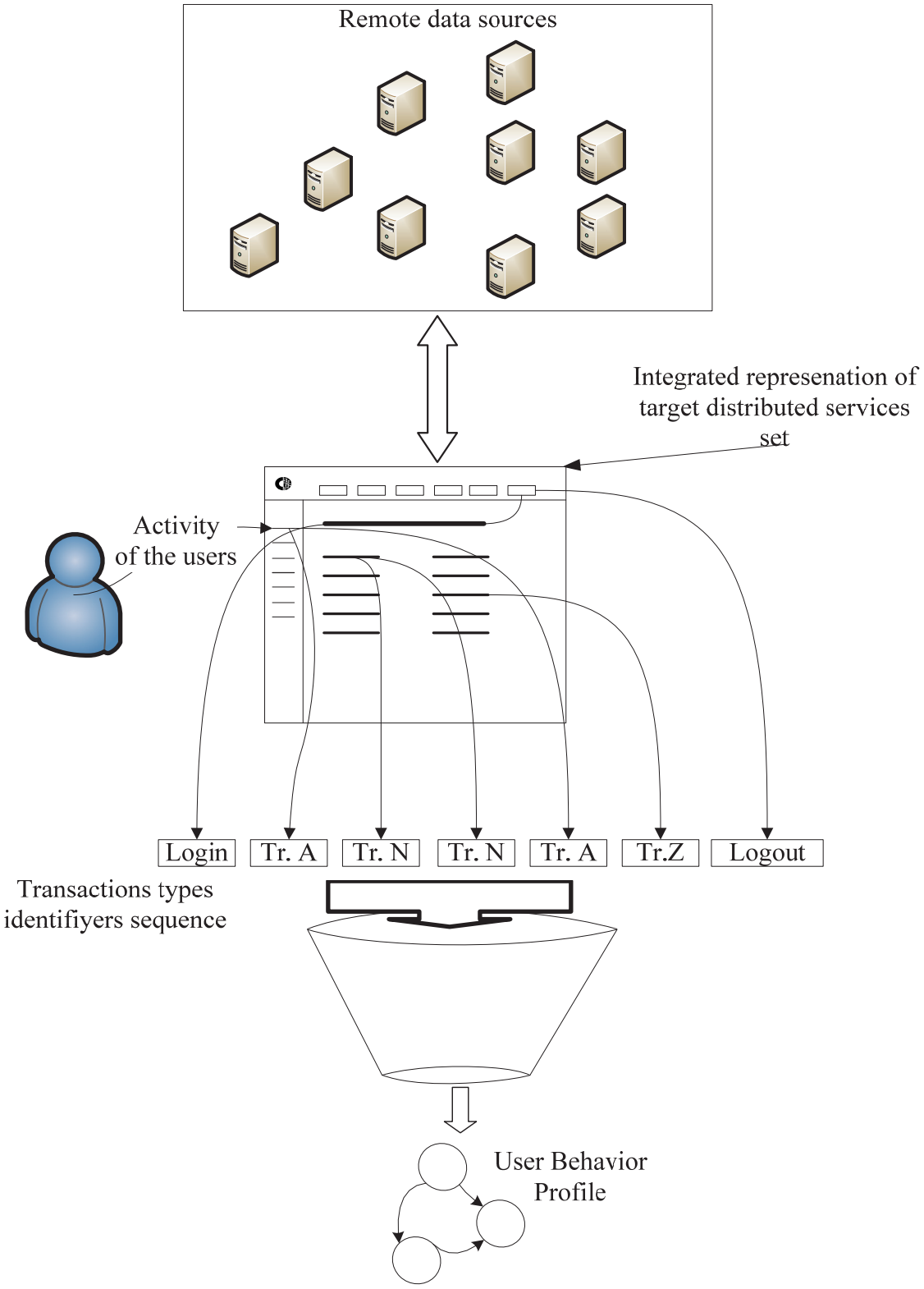


Fig. 2.2 General approach for UBP creation

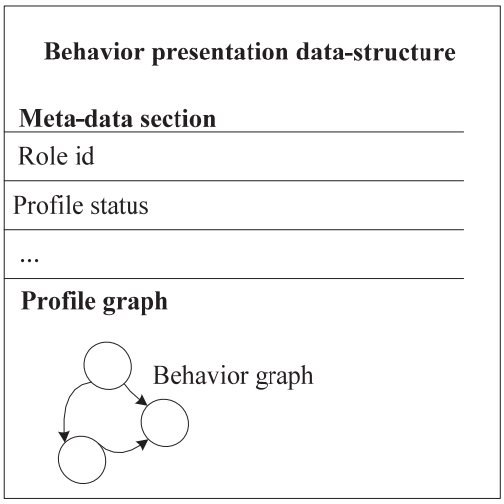


Fig. 2.3 UBP presentation data structure

2.2.2. Usage of the profile

This is main target mode of system functioning. The task is to use UBP that corresponds to current user for the calculation of anomaly level of each made by corresponding user transaction. Therefore, following activity proceeds every time user request any transaction on target system. This method is shown schematically on Fig. 2.4.

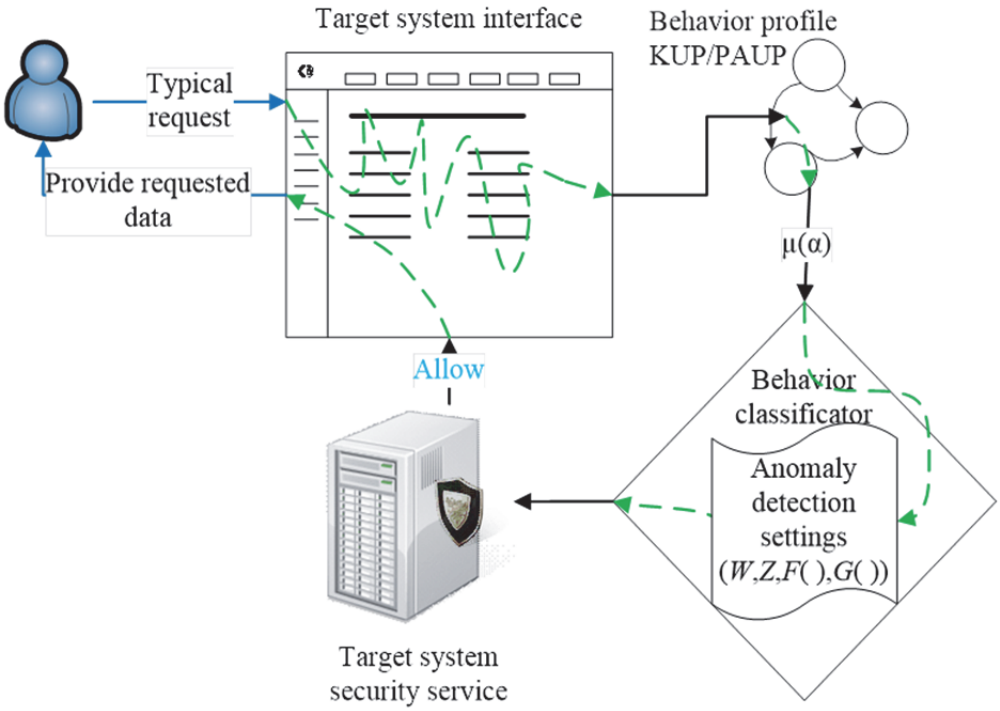


Fig. 2.4 General scheme of the approach used, in sace of "*typical*" behavior of user

The user inquires a certain transaction at target system. Internal module of security system inquires layer of such query anomaly at the module of anomalous behavior detection. The module of calculation of anomaly inquires UBP that corresponds to current user, applying the functions of anomaly level metrics receipt, the module receives necessary value. If anomaly level is higher than a certain threshold, (Fig. 2.5), anomalous behavior is considered as detected, if lower, a query is considered as typical for the user. Conclusion is send to basic module of security system, which decides how to take into consideration the obtained result.

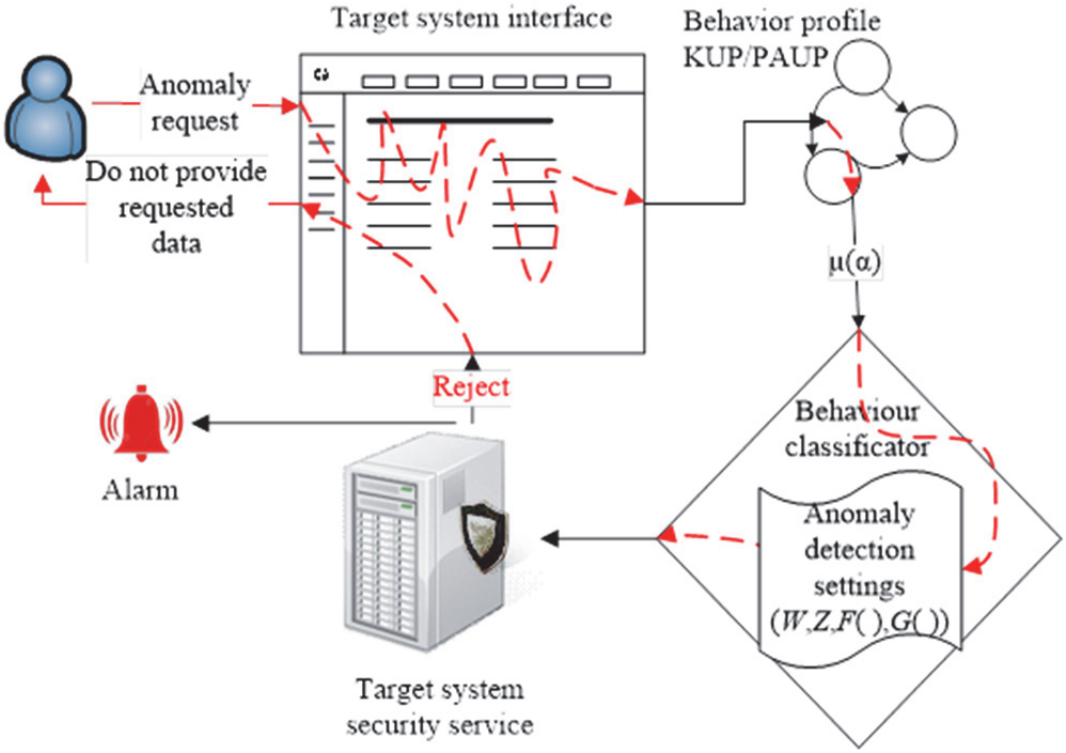


Fig. 2.5 General scheme of the approach used, in sace of "*anomal*" behavior of user

2.2.3. Testing of the profile

Testing of the created profile is required for validation quality of the obtained profile from the point of view of its ability to detect case of anomalous behavior.

At first, all existent set of suites is divided by subsets T and T_{an}, normal and anomalous traces. Further analysis consists of three stages:

- **Construction of test sets**. At this step, normal set T is randomly divided by two subsets, training T_{tr} and test T_{test}. Thus, test exceeds training on the amount of transactions.

- **Construction of classifier**. On the basis of training subset of traces T_{tr} User Behavior Profile is constructed.
- **Setting of parameters**. As for construction and estimation of the profile a few different variables are used, determination of their values affects final quality of classification.

Final UBP obtained from normal data of training selection has to divide correctly normal and anomalous data received from test selection for the analysis.

2.3. Formal description of base algorithm

Formal descriptions of all basic steps of base algorithm are shown below. As well as block, diagrams for each transaction are described.

2.3.1. Construction of behavior graph

In section 2.2.1 general training process of behavior profile is described; the aim of this paragraph is to describe in detail the process of construction of behavior graph based on history of the use of the system by user.

Special empty symbol *Ø* is added to the alphabet *Σ*. Initial value of window (*w*) is set. Initial state of Markov chain is determined as suite of length *w*, consisting of zero symbols, i.e. if *w*=3, then initial state will be: [*Ø, Ø, Ø*].

Two operations for traces are set:

- ***next*** (*σ*) (see Fig, 2.6) returns the first symbol of suite σ and moves σ on one position to the left, i.e. *next*(«*abcd*») returns and renews suite to «*bcd*».
  shift (*σ, x*) (see Fig. 2.7), moves suite *σ* to the left and adds the symbol *x* at the end of suite, i.e. *shift*(«*aba*», *c*) = «*bac*».

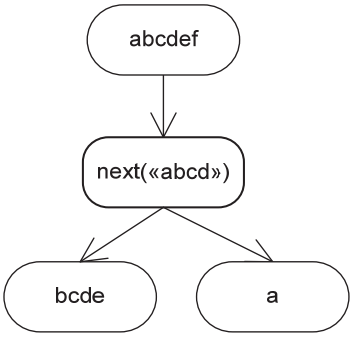


Fig. 2.6 Operation *next*()

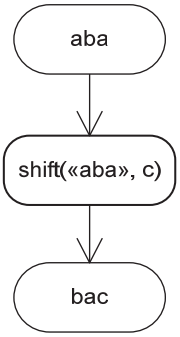


Fig. 2.7 Operation *shift*()

The cyclic process of Markov chain forming are presented on Fig. 2.8. Whey are consists of the following steps:

- suppose $c = \boldsymbol{next}(\sigma)$;
- set <next state> = **shift**(<current state>,c);
- increase meters for the state *<current state>* and transition (*<current state>*, *<next state>*);
- renew *<current state>* to the value *<next state>*.

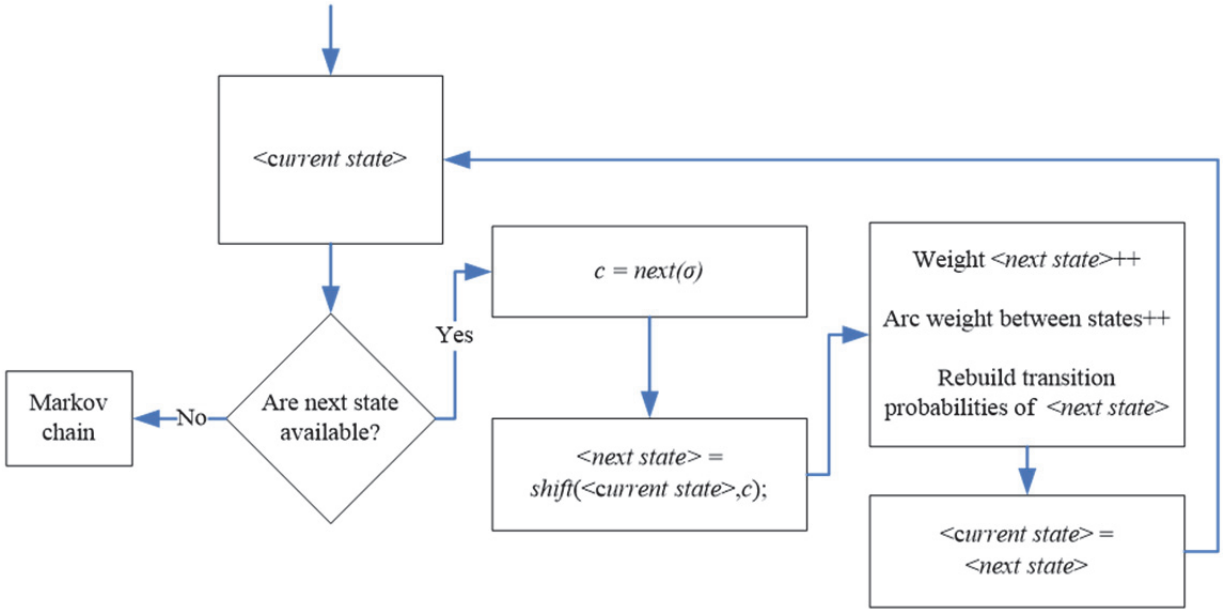


Fig. 2.8 Markov chain forming

Two values of meters for iteration are formed: value for current state and value for transition from current state to the next. The value of transition probability for an arc between nodes describing previous and current states is also set or updated.

2.3.2. Example of construction of behavior graph

As explanation of algorithm of graph forming on the basis of user behavior sessions the case of its construction on the basis of the following set of sessions is considered:

{a21,b12,c13|a21,b12,b22,c13|a21,b22,c13|a21,b22,c13|a21,b32,c13|a21,b32,c23,c33|a21,b32,c23|a21, a31,b32,c13|a21,a31,b32,c23,c33|a21,a31,b32,c23|a31,b32,c23}

Graphical visualization of behavior graph what may be constructed based on such set is shown on Fig. 2.9.

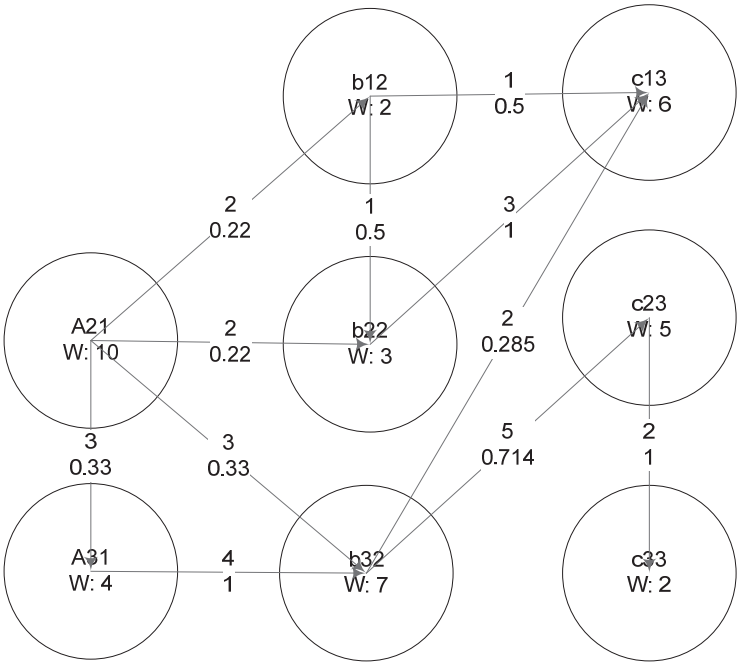


Fig. 2.9 Behavior graph construction based on data about behavior

2.3.3. Concept of anomalous level metrics

The calculation of the anomaly level for a session includes successive receipt of the anomaly values for each transaction, from which the analyzed session consists. Usually, the first transaction of session is logging in, but the last transaction is logging out.

Implementation of the procedure of calculation of anomaly level (Fig. 2.10) is performed for each transaction inquired by the user. Two internal variables X and Y are set; their values are tracked during work session. Initially (for the first transaction – *"logging in"*) they have the fixed value; it is allowed to use 1/0 or to calculate based on values of different internal parameters.

Each next step changes <*current state*>, adding the code of current symbol at the end and deleting one initial from description of current state.

On the basis of the constructed Markov chain containing the template of "*normal*" behavior for each step, metrics $\mu(\alpha)$ is calculated that determines status of current state and variables X and Y.

Two variants of functions-parameters X and Y values calculation are possible at each step (Fig. 2.10).

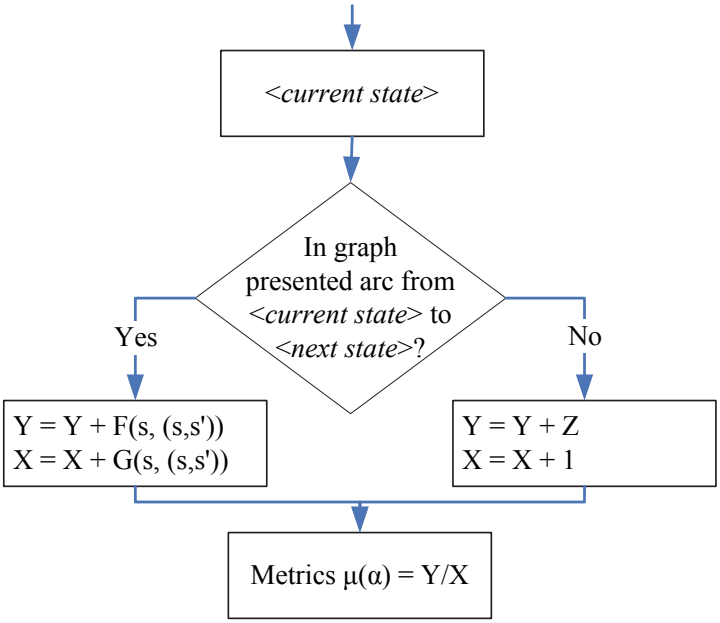


Fig. 2.10 Values of X and Y calculation

- There is a transition arc from previous state βi to target state $\beta i + 1$ at current graph. In this case X and Y are updated with the following functions−parameters:

$$Y = Y + F(s, (s,s'));$$

$$X = X + G(s, (s,s')).$$

- There is no transition arc from previous state to current state in current graph. In this case X and Y are updated with the following functions−parameters:

$$Y = Y + Z;$$

$$X = X + 1.$$

Finally value of metrics μ (α) is calculated that is equal Y/X.

Metrics $\mu(\alpha)$ shows, how current profile predicts suite α, i.e. the less is value, the more precisely Markov chain predicts suite α. As μ parametrized by functions F, G and number Z, then different selection of F and G will change behavior of classifier that adds possibility to be adjusted according to subject domain.

The classifier f constructed from metrics μ using following formula:

$$f(a) = \left\{ \frac{1, \mu(a) > r}{0, otherwise} \right\},$$

which is presented as block scheme on Fig. 2.11.

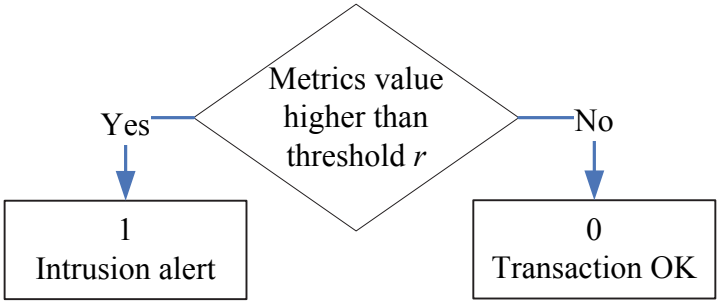


Fig. 2.11 Calculation of classifier

Here suite α will be classified as anomalous, if metrics μ for current transaction exceeds target threshold value r.

2.3.4. Approaches for metric value calculation

There are a few methods [50] for value of the anomality level calculation, using different F and G functions-parameters representation. Every time suitable method is selected depending on the features of current subject domain. As well as it is possible to calculate by several methods and to detect presence of anomaly even in one of results.

2.3.4.1. Frequency metrics

Using this metrics additional value, that is equal to maximum probability of transition to current node, is introduced:

$$P_{\max}(S) = \max_{s' \in succ(S)} P(s,s').$$

Using frequency-based metric the values of functions-parameters F and G are calculated using the following formulas:

$$F(s,(s,s')) = (P(s,s') \neq P_{\max}(s)),$$
$$G(s,(s,s')) = 1.$$

As block diagram the process of calculation of frequency metrics is shown on Fig. 2.12. It is important, what statistical significance of metrics is based on Markov property of common behavior graph.

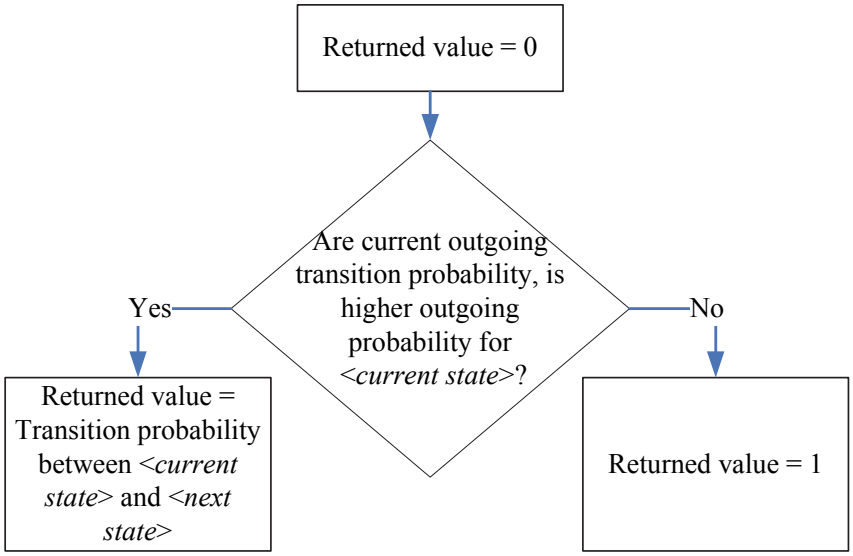


Fig. 2.12 Block diagram for calculation of frequency metrics

2.3.4.2. Probabilistic metrics

In case of probabilistic metric use, the following formulas used for the calculation functions−parameters *F* and *G*:

$$F(s,(s,s')) = \sum_{s_1 \in succ(s) \wedge s' \neq s_1} P(s,s_1);$$

$$G(s,(s,s')) = \sum_{s_1 \in succ(s)} P(s,s_1)=1,$$

where $succ(s)$ − set of all states, from which there is non−zero probability to get in the state *s*. As the graph of Markov chain has Markov property, the following condition is observed:

$$\sum_{s' \in succ(s)} P(s,s')=1.$$

Using this metrics, value F for each step (*s*, *s'*) is "*1 – probability of transition from previous to current node*". In other words, the less probability of transition, the closer to 1 will be a value of value *F* for this step, and, accordingly, final metrics that accumulates these values also will aim to 1 (Fig. 2.13).

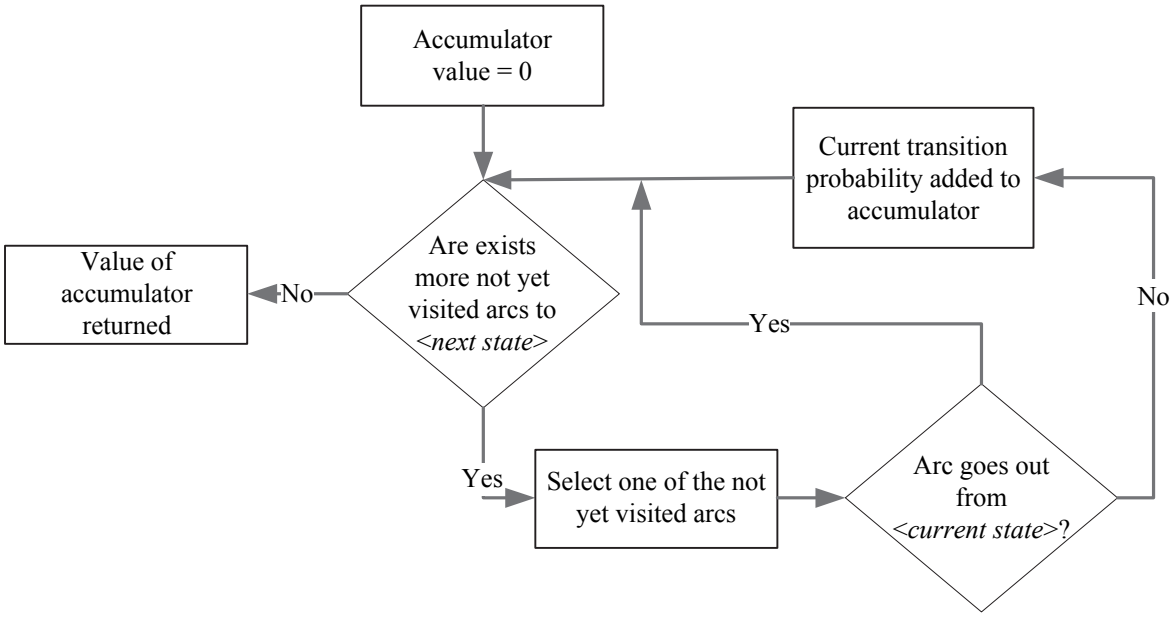


Fig. 2.13 Block diagram for calculation of probabilistic metrics

2.3.4.3. Local entropy−based metrics

This metrics calculates the value of entropy for transition from current to next state that is used as final value. In case of usage of this metrics, the value of local entropy for state *s* is introduced; it is marked as LE(s) and calculated using the following formula:

$$LE(s) = \sum_{s' \in succ(s)} -P(s,s')\log(P(s,s')).$$

As a result, the functions−parameters *F* and *G* are determined as follows:

$$F\big(s,(s,s')\big) = LE(s) + P(s,s')\log(P(s,s'));$$
$$G\big(s,(s,s')\big) = LE(s).$$

For transition, (s,s') expression $LE(s) + P(s,s')\log(P(s,s'))$ is calculated as: $\sum_{s' \in succ(s) \wedge s_1 \neq s'} -P(s,s_1)\log(P(s,s_1))$, that is similarly to probabilistic metrics. As a result value *LE*(*s*) determines residual local entropy of state s after deleting transition (s,s'), or decrease of entropy after deleting this arc. Block diagram for calculation local entropy of only one node *LE(s)* is shown on Fig. 2.14.

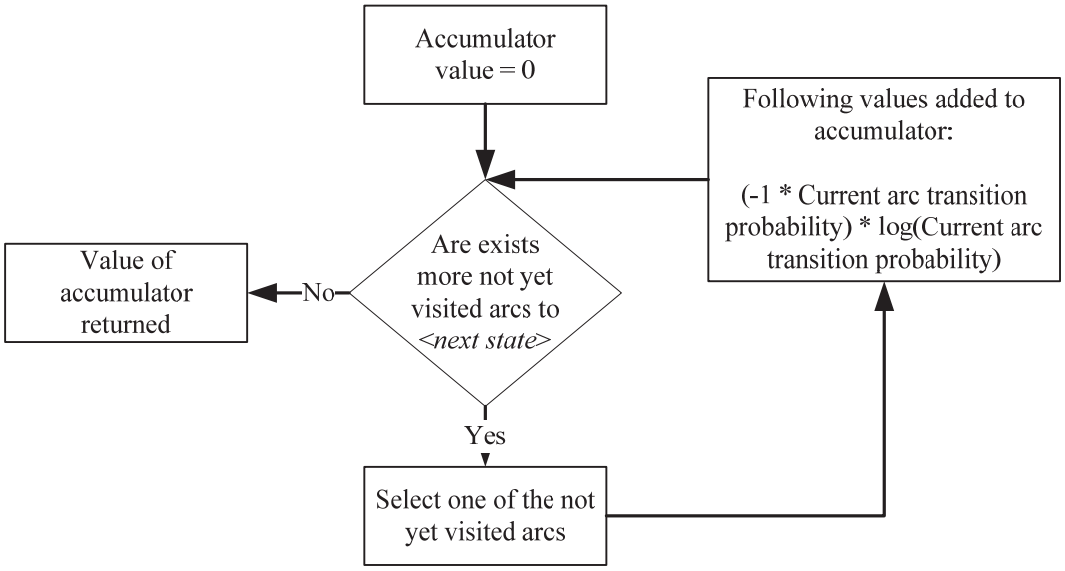


Fig. 2.14 Obtaining of local entropy of node

Block diagram for calculation of a decrease in entropy after deleting target arc is shown on Fig. 2.15.

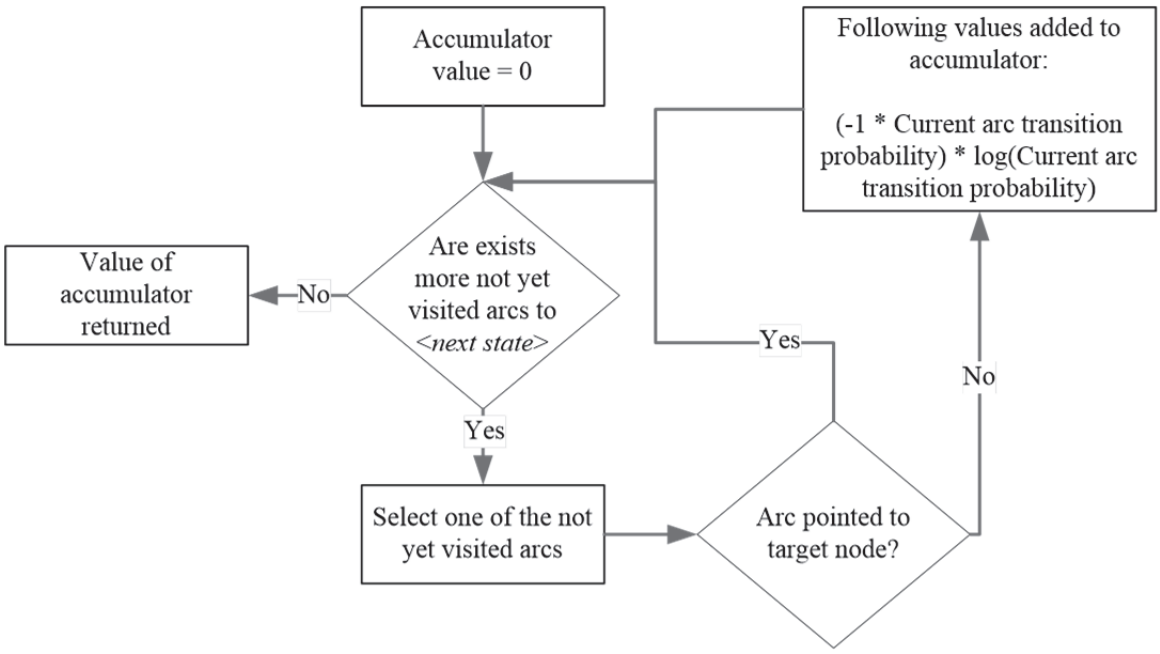


Fig. 2.15 Entropy after deleting target arc

2.3.5. Parameters of algorithm and their impact on quality of result

Because many parameters are used in the construction of classifier, determination of their values impact on quality of resulting classifier. In addition, it is important to determine the values of metrics of classifier processing power, which are calculated on the basis of both initial sets of traces T and T_{anomal}.

Quality of the created classifier depends on initial training set T_{tr} and the following parameters:

- size of window w;
- type of functions $F(s,(s,s`))$ and $G(s,(s,s`))$;
- value of parameter Z;
- threshold r.

Possible variants of calculation F, G and Z are considered above; w and r are considered below.

Selection of size of window w may be disputable, because small values contain not enough information, but large value is adjusted exactly to training set (*over fitting* [41] − a classic problem of re-training). If to perceive metrics (σ) as measure of difference between Markov chain and analyzed step, i.e. the less it is, the better fragmentation, then discrepancy for all trace in all training set $D(T_{tt})$ can be obtained using the formula: $\frac{\sum \sigma \in T_{tt} \mu(\sigma)}{\sum \sigma \in T_{tt} |\sigma|}$. It is equal to relation of sum of metrics values for all steps to the sum of the values applied to each node of Markov chain. Similarly, we calculate discrepancy of trace to anomalous $D(T_{an})$.

In this case, size of window w is increased, while difference $D(T_{an}) - D(T_{tt})$ is higher than specified value, i.e. the classifier divides these two subsets.

Regarding to parameter r that is a level, in which the value of metrics for a step is determined as anomalous, in this case it is important to use optimal value. In this research, its value is specified so that the number of trigger T_{an} approximately corresponded to training set with the templates of anomalous behavior.

2.3.6. Concept of behavior profile "*complexity*"

The concept of "*complexity of UBP*" allows estimating formally the number of nodes and connections necessary for presentation internal UBP graph that keeps data about user behavior. Having such estimation developers and researches may make planning of hardware

resources what whey will have needed to realize system based on approach described. Are single server will be enough or initially should be developed multy-server architecture.

To estimate "*complexity of UBP*" we will introduce additional designations:

G – graph describing current UBP;

G_{nodes} – the number of graph nodes;

G_{arcs} – the number of graph arcs;

$bytesize(item)$ – function returning the number of bytes required for presentation one programmatic element in memory;

$metadata$ – additional data used for programmatic presentation of one UBP;

Σ – an alphabet, full set of all possible types of transactions in target system;

w – size of window.

The value of "*complexity of UBP*" is multiplication of the number of all nodes of internal graph by the number of all arcs:

$$S(G) = G_{nodes} * G_{arcs}.$$

Maximum *complexity of UBP* $S^*(G)$ describes a limiting case, when the number of nodes is equal Σ and each node is connected by arcs with all other nodes. In this case estimating of *maximum complexity of UBP* will be equal to $O(S^*(G)^w)$. This value can be very large, but it is a limiting case and in real terms each user will be interested only in limited subset C^* of *C*, such as $C^* \ll C$.

The value *maximum complexity of UBP* allows calculating the volume of operating memory required for presentation of behavior profile. In general case maximum value will be equal to:

$$RAM^* = G_{nodes} * bytesize(node) + G_{arcs} * bytesize(arc) + G_{nodes} * bytesize(metadata).$$

2.3.7. Estimating of complexity of base algorithm classification

Basic time delay at this stage is work with a graph. For the calculation of metrics, it is necessary to analyze full trace of the user within the framework of current session. The user in order to get in current state can pass all nodes of graph that provides estimating of complexity $O((G_{nodes})^2)$ that is a relatively large value, because the complexity of final UBP cannot be predicted initially.

However usage of Markov chains allows not to analyze all previous trace of the user, and only two nodes − *current* and *next* (inquired transaction). Let`s assume that pr is the number of nodes with non−zero probability of transition to current state, *cur* − a node describing current state and *nxt* − a node describing the state after implementation of the inquired transition. All these values are much less than N, i.e.:

$$pr + cur + nxt << N.$$

It provides constant complexity of algorithm $O(pr + cur + nxt)$, when it is necessary to get only data about all previous nodes of the first layer for current state, node of current state and node inquired for a transition.

Other operations of calculation metrics also have not large values of complexity estimating, because also operate only current state nodes and do not contain deeply nested loops and difficult calculations what may require unpredictable lot of iterations what are takes system resources.

2.4. Advantages and lacks of base algorithm

As any algorithm, the considered basic algorithm has its weak and strong sides. The list of basic advantages and lacks is shown in Table 2.1. In spite of the fact that the number of lacks is more that the number of advantages, advantages are more significant, especially taking into account the existing limitations and requirements to the system.

Having data about typical behavior of user (or class of the users united by same *role* in target informational system), it is possible to create a profile that will allow estimating the degree of compliance of each new step of user with typical behavior in the system. Accuracy of results and presence of errors in security system depends on the structure of created profile and methods of its analysis. For one target system, even small anomaliy presence will be reason for alers, in others user may be allowed to make some not very typical requests without informing security module.

Depending on settings (for example, required level of accuracy) and specific of subject domain, the system of analysis can characterize a transaction as anomalous, as well as “normal”. The result of classification can be send to the expert – person for final decision-making.

Table 2.1

Advantages and lacks of base approach

Advantages	*Lacks*
• Possibility to update behavior profile automatically in case of new data about the actions of user. • The structure of algorithm allows the easy implementation of processing large number of profiles at the same time. It gives an opportunity without considerable effort to increase the system processing power, if it is required increasing the amount of served users. • Base algorithm does not require a lot of input data for its work, minimally sufficient evidence of three types: o unique identifier of user; o identifier of its role; o identifier of made transaction. • Base algorithm can be easily extended with new types of metrics, what are better works for specific of target problem domain.	• Construction of UBP – is difficult and time-consuming task. • In the task of updating UBP a significant problem is determination of criteria for activation of user profile updating. • Data about behavior of different users united by same role inside of target system have different statistical characteristics; it makes a final model too general. • Determination of limit values for UBP, which exceeding is considered as anomalous behavior (intrusion), depends on the specific of current subject domain and cannot be formulated in general. • Possibility to hide illegal actions, combining "*good*" and "*bad*" queries. • Additional load on the system, arising out of functioning the system of analysis each transaction in real time. • Presence of confidential information in data of the system.

2.5. Improvement of base approach

Efficiency of base algorithm, solving the tasks in real life, can differ significantly, depending on the specific of subject domain. The task is to improve base method with the purpose to increase its efficiency. The task to calculate efficiency is considered in detail in chapter 3. To improve the level of the anomaly detection and to eliminate basic lacks of basic approach, it was suggested to change general profile of class of users with personal profile.

2.5.1. Personal adaptive profile of user behavior

One of basic lacks of profile for role of users is that behavior of plenty of users (Fig. 2.16), even by same role inside of target system can have large uncertainty. As a result, UBP constructed on the basis of such data will describe behavior of some general representative of role. Such general profile describes behavior of each user of role similarly.

To increase accuracy of analysis of user behavior it is assumed to use Personal Adaptive UBP (PAUBP). Such profile will include information about a specific and features of behavior of one-target user (Fig. 2.17).

Profile adaptivity is provided by the procedure of **continuous updating of profile** based on user activity. General rule of updating PAUBP is the following:

When a user completes session, with the condition that anomalous behavior is not detected, data of session are used for updating the appropriate PAUBP.

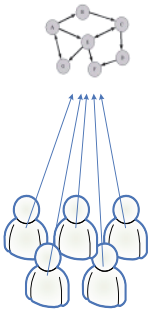


Fig. 2.16 Common profile

Fig. 2.17 Personal profile

Obvious that such method will allow getting a profile that more precisely describes the features of the appropriate user behavior. The level of increasing the efficiency between UBP and PAUBP described in chapter 4.

Basic **advantages** of PAUBP are the following:

- higher accuracy of detection anomalous behavior in comparison with UBP;
- continuous adjustment according to target user behavior;
- possibility of flexible correlation of different profiles to one user.

Lacks are the following:

- proportional increase of load on hardware of anomalous behavior detection module because of replacement one general UBP for the group of users unified *by common role*, with personal profiles for each user;
- forced additional training PAUBP is required in case of significant changes of the target system.

In comparison with detected lacks, advantages are perspective; therefore, the method to increase efficiency of UBP will be transition to the structure of PAUBP.

2.5.2. Dynamic threshold of anomaly level

The method used in base algorithm, when level of anomaly is determined by fixed value of constant r, in general case is not the best. By increase of general value of metrics, its level can exceed the set threshold and then even typical behavior will be considered as anomalous. As a result, we get the following situation:

- at presence of constant "*normal*" behavior, the stationary values of the anomaly metrics is provided;
- then in case of receipt of anomalous queries, the level of metrics decreases;
- then many "*normal*" queries follow, which provide the stationary values of the anomaly metrics again, but already at lower level.

In form of chart, description of such situations is shown on Fig. 2.18. After increasing level on anomalious requests, value of metric stabilizes on new level during the "*normal*" activity. To solve this problem, the method with usage of **dynamic threshold** − r^* is offered. In this case, value r^* is calculated every time, when a response returned by the module is calculated. In other words, r^* − is the appropriate value of current dynamic threshold for every calculation of anomaly level for received query.

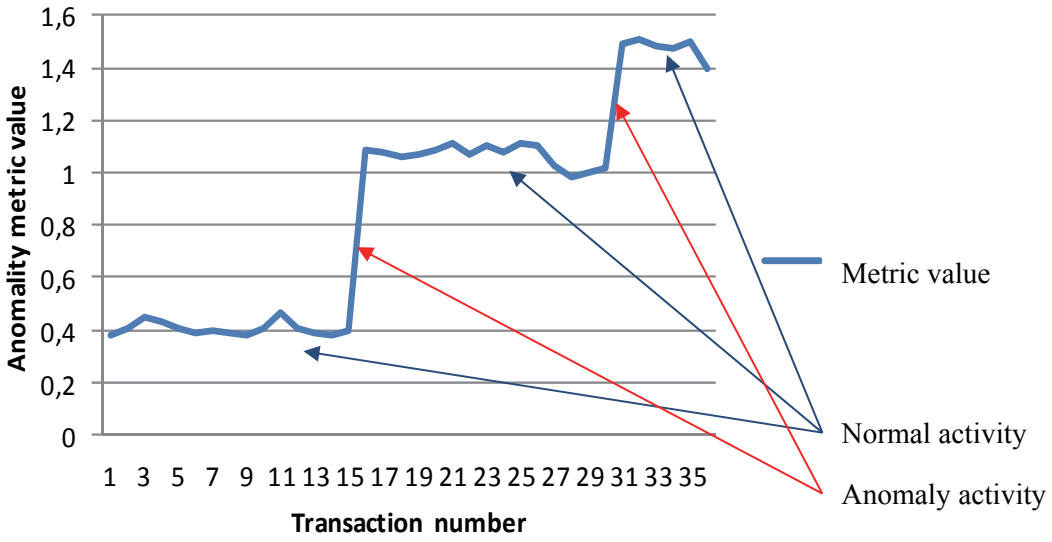


Fig. 2.18 Levels of the metric values stabilization

Let`s remind that the value of anomaly level of typical behavior aims at 0, i.e., $A(normal) \to 0$, accordingly the value of metrics of anomalous behavior aims at some value Á > 0, $A(anomal) \to$ Á. The value Á depends on the features of subject domain, as well as on the specific of current user behavior. In some systems, insignificant increase of value of anomaly level will be enough for alarm, but in other systems, the user has to implement many untypical actions in order to activate alarm.

In this case, basic criterion for the selection of value r^* is level of increase value of metrics during last *n* queries. *Level of increase* in this case is difference between the value of metrics before the implementation of previous steps and after.

Introduces the concept of $2(s|n)$ – level of increase of the metrics value in transition from $s_{i-n}\ to\ s_i$:

$$2(s|n) = (\sum_{i=n}^{0} A(S_{i+1}) - A(S_i)) - (\sum_{i=n}^{1} A(S_{i+1}) - A(S_i)),$$

where *n* – the number of previous steps analyzed in the calculation of current threshold of anomaly, the value is set at the stage of setting the method;

i – index of current integrated state in the loop.

Then dynamic anomaly threshold for the current step $-r^*$, be equal to difference between levels of increase for current and previous steps:

$$r^* = 2(s|n) - 2(S_{i-1}|n).$$

However, final response of classifier will be not constant value of metrics, higher which behavior is considered as anomalous, but other constant *r`* − maximum possible level of increase for normal state:

$$f(a) = \left\{\frac{1, r^* > r`}{0, otherwise}\right\}.$$

Dynamics of change anomaly threshold value for some test behavior of user is shown on Fig. 2.19.

In case if the current increase of the anomaly level is more than specified level, the upper threshold of "*normal*" level is limited dynamically. What allows to build flexibe rules of "*anomaly*" detection alarm activation, based on specific of target problem domain.

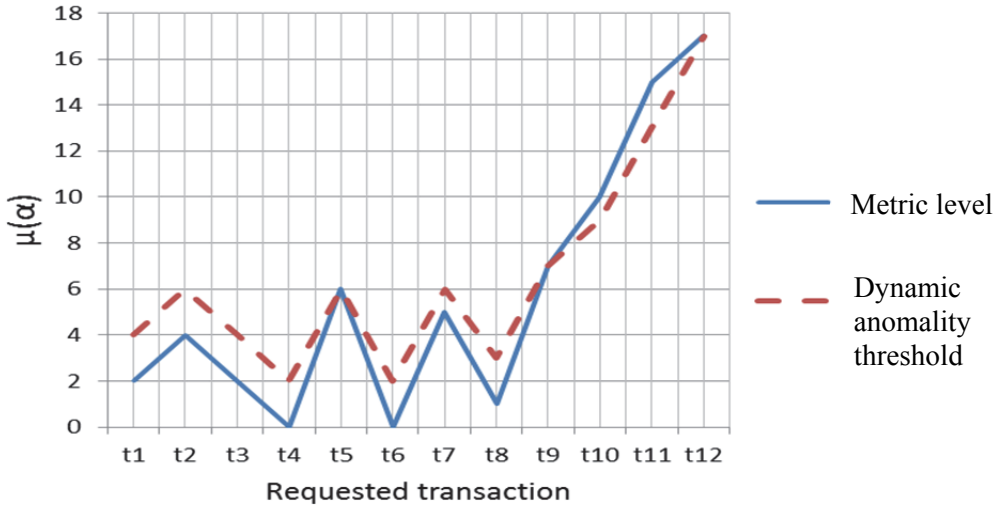


Fig. 2.19 Dynamics of anomaly threshold change

2.6. Summary

This chapter describes basic approach to solving user anomalous activity detection tasks. The algorithm is based on the use of Markov chains.

Common description of used basic approach to construction (training) and use of User Behavior Profile provided understanding of features of used basic algorithm without the necessity to investigate formal description in detail. The complete formal description of basic algorithm provided availability of its detailed description in the case of necessity.

- Possibilities of the use of Markov chains are shown as data structure appropriate for description of user behavior as the oriented graph. Markov property imposed on the graph as limitation reduces the amount of processed information, receiving request for the analysis of the anomaly level, because allows to analyze not whole graph, but only node describing the current behavior and transaction inquired by user.
- Main advantages and lacks of basic algorithm are specified. Approaches for improvement of basic algorithm are offered − the use of dynamic threshold of the anomaly assessment, as well as personal profile of description of user behavior.
- The offered method of dynamic threshold allowed more exactly in comparison with the fixed threshold to determine the level of anomalous behavior.
- The offered approach to increase efficiency of anomalous behavior detection, replacing general User Behavior Profile with Personal Adaptive Profile of User Behavior, allowed preventing majority of detected lacks of basic algorithm.

3. ESTIMATION OF EFFICIENCY OF USER BEHAVIOR PROFILE

Base algorithm uses many sets of transactions for creation UBP. For this purpose users by same role are selected (role determines a range of tasks and interests of the user in the system). Final profile includes summarized data about behavior – General UBP (GUBP). In the process of creation of GUBP, data are averaged; so the profile becomes less sensitive to the changes of the appropriate person behavior. There is the following assumption – it is possible to increase efficiency of detection anomalous behavior, creating *Personal Adaptive Profile* that is trained on the features of the appropriate person behavior. Such profile will be more sensitive to the changes of the appropriate person behavior.

In this chapter, theoretical basis of the methods for estimation of efficiency of information systems is considered. Usage of probabilistic characteristics allows usage of statistical indexes for comparison of the systems. Methods that use the value of entropy are used for estimation of the system from the point of view of information theory. Alternative methods based on other methods for estimation of efficiency are possible too.

3.1. General methods for estimation of efficiency information systems

There are different methods for estimation of efficiency of information systems [98, 19]. Depending on the specific of the researched task, general efficiency indexes (*reliability*, *authenticity* and *safety*) or set of personal indexes (characterizing *pragmatic*, *technical*, *technological* and *operating efficiency* of the system) are used. Depending on the selected basic criterion and the system of indexes, different mathematical methods for estimation of efficiency of difficult systems are used:

- methods of information theory;
- methods of probability theory.

3.2. Probabilistic characteristics for estimation of quality of user behavior profile

Using probabilistic method, the estimated object is considered as a "*black box*" [7]. Comparison of objects is implemented by implication, by comparison of statistical parameters.

As applied to UBP, such method can be the following – to compare characteristics of metrics issued by the profiles trained on different selections. Significant difference in training selection has to be shown on resulting quality of detection anomalous behavior.

Using such method, profiles can be considered from the point of view of their difference from general profile of class, as well as of statistical properties, when dispersions and values of standard deviation of sets of values of metrics from different profiles are compared.

Interest levels are schematically shown on Fig. 3.1. *A* in this case is a full set of all possible types of queries in target system. Then *A*' is subset of interests only part of users united by same role inside of target system; *A*" is a subset *A*' that shows specific interests of a certain user. Obviously that a profile that shows interests of group *A*', will have other statistical characteristics in comparison with *A*'.

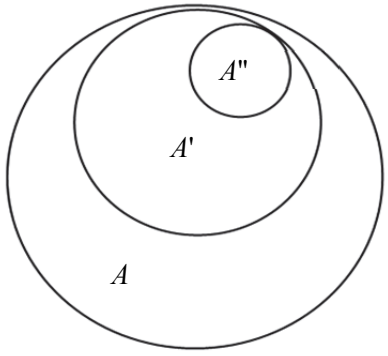


Fig. 3.1 Interest levels nesting

Presenting gradation of interests in such way, it is possible to determine **probabilistic characteristics for estimation of quality of UBP**. In this case, quality of UBP can be estimated by expression "1 - *probability of erroneous conclusion of profile*" [50]. Accordingly, behavior such profile is better, which has less probability of error of both types (when "*anomalous*" behavior is interpreted as "*normal*", or vice versa):

$$Q(\text{UBP}) = 1 - \left(\text{p}\left(Err_{fpr}\right) + \text{p}\left(Err_{fnr}\right)\right),$$

where $Q()$ – general concept of profile suitability;

$p\left(Err_{fpr}\right)$ – probability of the first type error (False positive error rate);

$p\left(Err_{fnr}\right)$ – probability of the second type error (False negative error rate).

Usage of such method gives an opportunity to show efficiency of GUBP in comparison with PAUBP, when framework of general profile allow to implement a lot of various actions that will be perceived as "*normal*", while personal profile describes more limited subset, and probability to access for off-site person is less.

Let`s consider the last statement in detail. On Fig. 3.2 it is shown, how users united by same role inside of target system may have different subsets of personal interests. If the user A''_1 logs in the system with account of the user A'_n, starting to use it, how he does it usually, every his action will be interpreted as anomalous, in spite of the fact that they have same role inside of target system. Final dispersion of values of interests for the class of users will be always higher, than dispersion for a certain person.

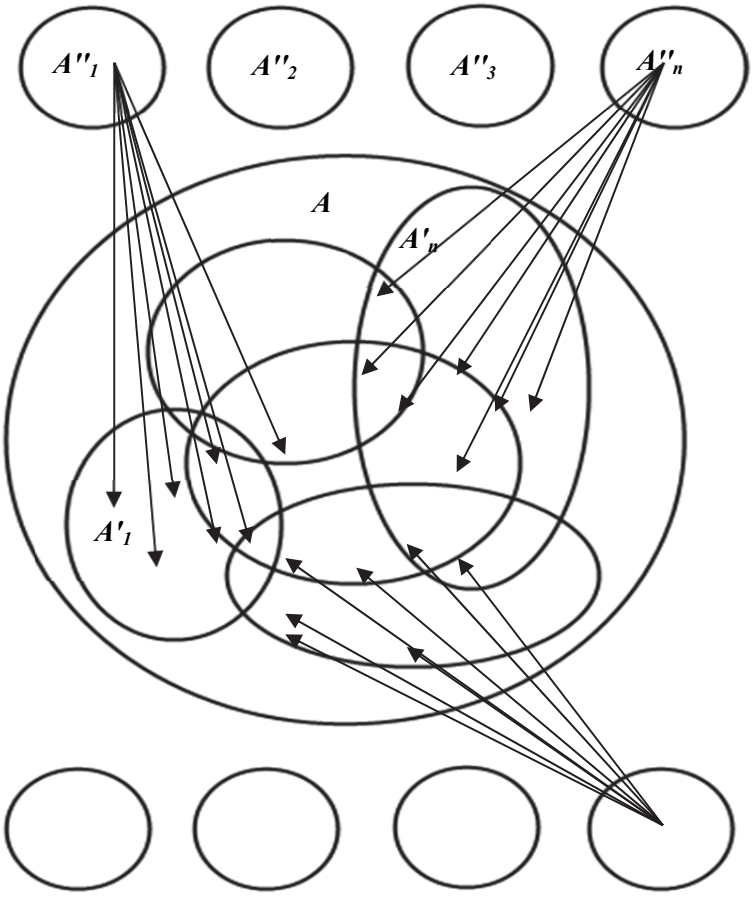


Fig. 3.2 Users different interests belonging to the same role

3.3. Methods of information theory−based on entropy

Basic characteristic for estimation, using information criteria, is *information carrying capacity* [19, 90, 4, 34]. The higher it is, the more effective the system operates, the more information it receives from every report. Therefore, it is enough to compare efficiency indexes for detection anomalous activity using GUBP and PAUBP. In this case, unit of measure of estimation is entropy of every report on the action implemented by the user. That gives an opportunity to compare these indexes using different methods as described two branches on Fig. 3.3, or using the formula (3.1).

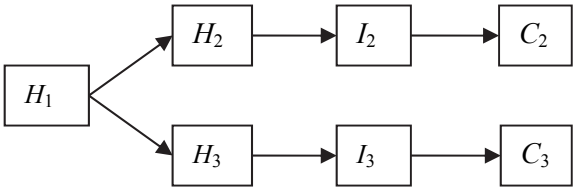


Fig. 3.3 Efficiency comparison using entropy difference

On Fig. 3.3 H_1 − initial entropy of the system without using UBP;

H_2 − entropy of the system after introduction GUBP;

H_3 − entropy of the system after introduction PAUBP;

I_j − average amount of information, replacing one profile by other:

$$I_2 = H_1 - H_2$$

$$I_3 = H_1 - H_3;$$

$C_i = I_j / \tau$ − information carrying capacity of the system using the profile j for analysis of transaction τ.

As a result, the system with larger value C will be more effective accordingly following formula (3.1):

$$E = \begin{cases} C_2, & C_2 > C_3 \\ C_3, & \text{otherwise} \end{cases}, \tag{3.1}$$

where E − the most effective profile.

As the researched system has resulting amount of the states, for estimation of its entropy a base formula is used for the systems with discrete states:

$$H(X) = -\sum_{i=1}^{n} p_i \cdot \log p_i \ .$$

3.4. Alternative information criteria – the Bongard method

There are other methods for usage of entropy for estimation of quality of information systems. For example, in [12] method is considered, when, as a basic information index of efficiency of the system is average number of steps to achieve a goal:

$$n = \frac{Initial\ system\ entropy\ value}{One\ measurment\ entropy}.$$

Such *n* determines average time of solution of task, i. e. productivity of the system. The level of efficiency of integrated node in this case increases the value of *degree of utility* of every input message for a recipient; it promotes decrease of degree of uncertainty of task, consequently, decrease of amount of steps required for decision. However, such method can be used only if all basic parts − decided task, initial state of solving algorithm and features of decoding algorithm − are formally determined.

As a measure of difficulty of current task, logarithm of the amount of iterations necessary for solution of task by solving algorithm is used. It is necessary to take into account that incoming reports affect the sequence of iterations and consequently, their amount necessary for solution of task (i.e. its uncertainty).

3.5. Summary

Review of basic assessment methods of user behavior profile efficiency allowed successful planning its own experimental system, specialized on assessment of properties of User Behavior Profile, General User Behavior Profile and Personal Adaptive Profile of User Behavior. In addition, researches described in this chapter allow determining major characteristics affecting efficiency of User Behavior Profile that in turn impacts on types and purpose of experiments conducted during research.

- Review of general approaches for assessment of efficiency of information systems allowed determining approaches actual within the framework of this research.
- The use of methods of information theory within the framework of the current task allowed estimating properties of behavior profiles, not taking into consideration features of their realization.

- The use of approaches for information transfer allowed estimating efficiency of behavior profiles, using the metrics based on entropy.
- The considered alternative method to estimate efficiency of information systems provides interesting ideas within the framework of solving the current task-comparative assessment of efficiency of Personal Adaptive Profile of User Behavior and User Behavior Profile.

4. EXPERIMENTAL SYSTEM'S DESIGN AND PROGRAMM REALIZATION

In order to show that theoretically operable system can solve real tasks, programmatic platform has been created for the realization of experiments. All operations described in theory are realized using Python [59] and PHP programming languages, the infrastructure for the realization of experiments has been created.

For successful creation of experimental system, it is necessary to describe the basic features of subject domain. For creation of the system, it is important to take into consideration limitations imposed on available resources. Then we have to create the structure of created system, to select and to ground used technologies for its realization. To provide effective development of experimental system we have to select the best programmatic tools for realization of the system. Specific terms what are used in this chapter:

- XML – markup language that defines a set of constraints and rules for presenting documents in a format that can be easy readable as by human as by machine [28];
- JSON – an open standard format that uses plain text to transmit data objects, typically consisting of attribute/value pairs (but not limited by it), between different envelopes [100, 91];
- WEB – is a set of interlinked hypertext documents that are accessed via the Internet;
- eHealth – type of system what moves specific medical organizational operations into electronic form [64];
- EHR (Electronic Medical Record) – patient's medical history, presented and stored in digital format [36, 75].

4.1. Features and limitations of subject domain

In this chapter features specific for the target information system are described, which affect the possibilities of introduction of described approach. Some details of architecture can require small changes in order to introduce the created system; other changes can make introduction impossible.

As basic features of subject domain can be specified the following, most meaning for anomaly detection, characteristics.

- It is considered that each type of Information System (IS) user has typical behavior during work within the framework of IS.
- As feature of intrusion, significant (at least for set value) difference between metrics value of user current behavior and typical behavior is used.
- Amount of time available for the module of detection anomalous activity for the analysis of each transaction is limited by the requirements of Quality of Service of the system. Typically, result should be presented in real-time mode.
- Programmatic architecture of target IS requires maximum isolation of each module, i.e. the module is perceived as black box receiving only required data and returning a decision.
- Target IS operates simple, as well as sensitive data; all data require different methods for their processing.

As basic features of anomalous activity detection method based on Markov chains, it is possible to specify the following:

- The calculated metrics of user behavior are statistically reliable. For example, for the calculation of value of metrics different methods can be used: *probabilistic*, *frequency*, *entropy-based*.
- In case of usage of adaptive profiles possibility to realize expansion of algorithm for continuous training appears, when Markov chain lying based on every profile is adjusted according to the features of the appropriate user behavior, increasing accuracy of estimation of his behavior and accordingly correctness of each transaction analysis. Therefore, accumulating data, efficiency of the system increases.

When building UBP based on historical data about system usage in past, two methods for forming internal Markov chain are supported:

- Training with a teacher, when records about already made transactions are given. All set of data for training are data that are estimated positively.
- Training without a teacher, when every next transaction made by the user is estimated in real time, i.e. the system is trained based on current activity of user.

For work, the method needs only the identifiers of inquired transaction and user that made this transaction; all other realization is independent and hidden. Output data is only a response – 0÷1 an anomaly level calculated.

4.2. General structure of the system

For realization of experimental research of developed theoretical method, programmatic system has been created. General structure of the system is shown on Fig. 4.1. Basic entry point shows available menu for implementation of experiments. The system is divided into logical modules. Each module implements the part of required functionality.

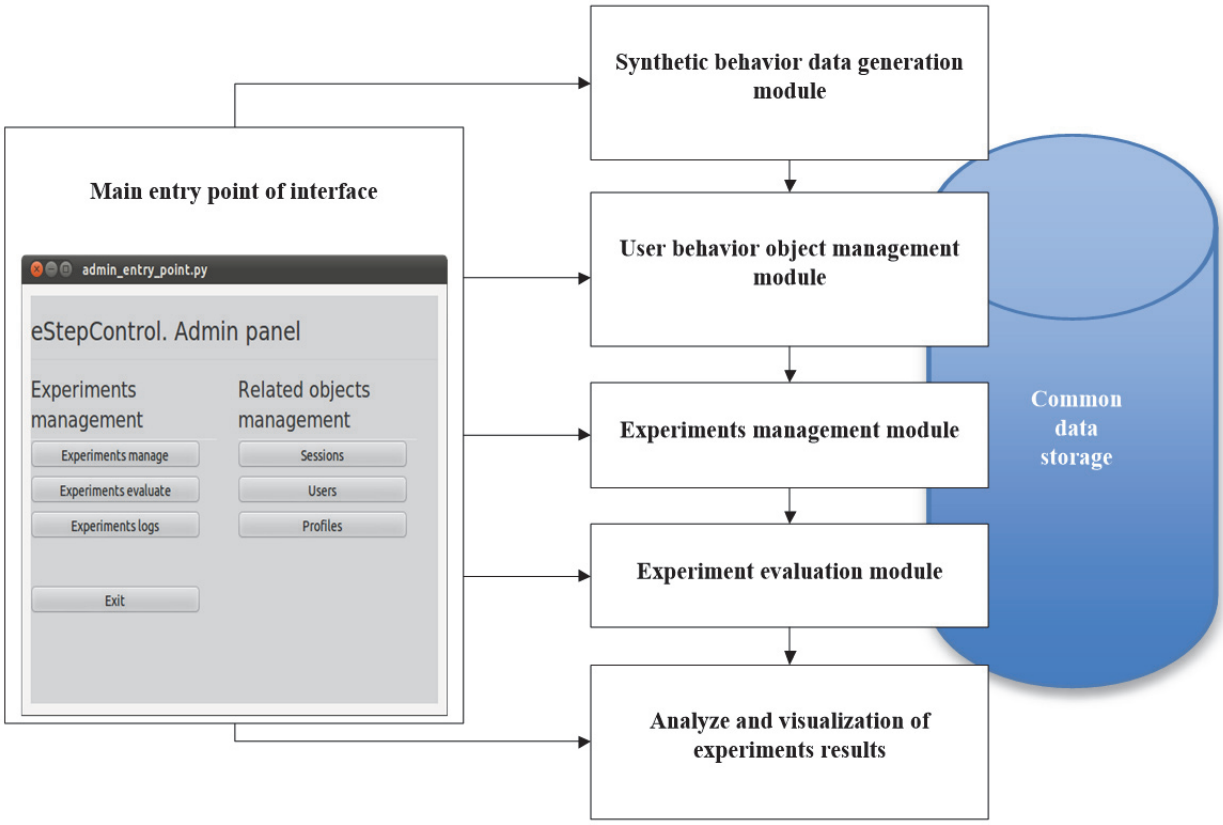


Fig. 4.1 General strusture of the experimental system

Based on Object Oriented Programming (OOP) approach, each module realized as single class presented in his own single file. Each of modules realized are described further.

4.2.1. Session generation module

Module allow to genereate users behavior sessions with different required characteristics needed. The generation of behavior sessions can take place in two modes: manual and automatic. The form for generating behavior data included in training sessions for profiles is shown on Fig. 4.2.

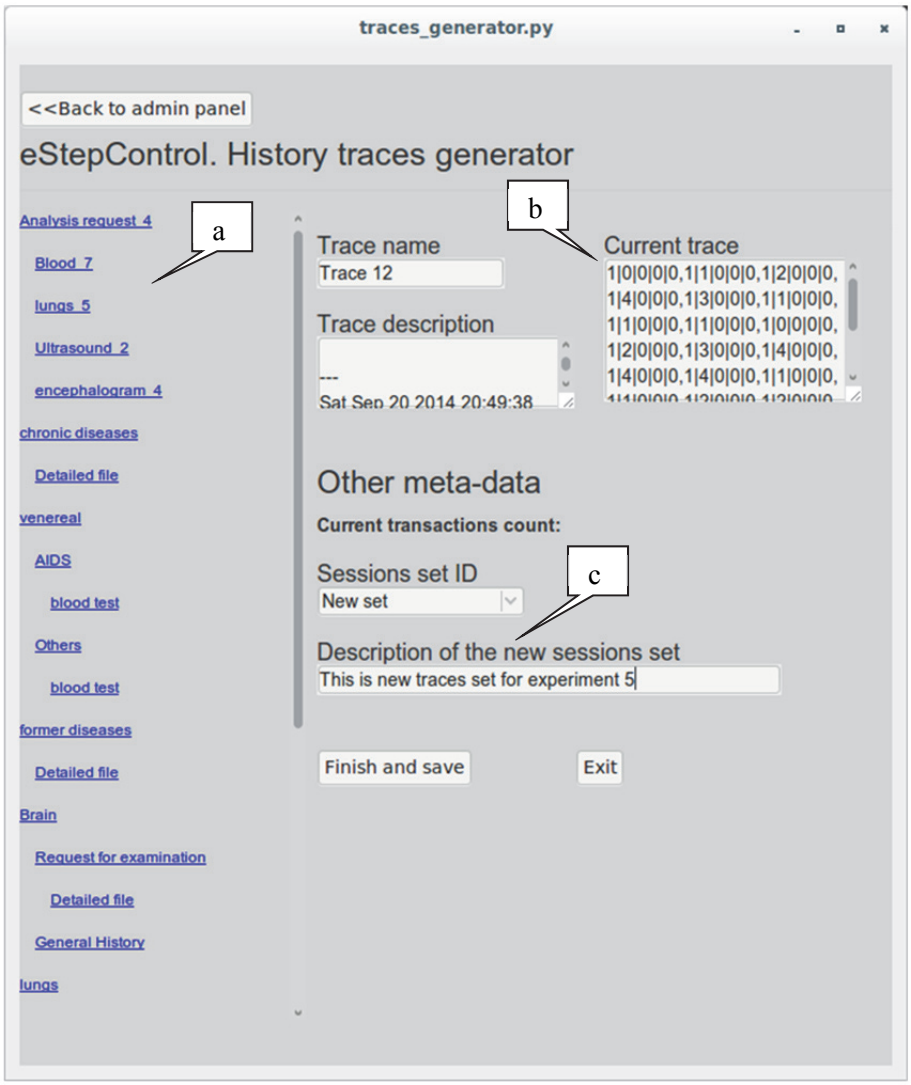


Fig. 4.2 Form for behavioral data generation in manual mode

On this form following followig functionality are available for person who administrate session generation:

a) manual navigation by the categories of user interests;
b) on the basis of developed navigation it's created a path as a vector of identifiers of inquired categories;

c) the created session can be included in the existing series of sessions or new series can be created.

4.2.2. Behavior profile management module

This module provides functionality for creation, training and deleting of UBP programmatic copies (Fig. 4.3). For training of profile based on new behavior, data following steps should be implemented:

a) choose the target profile;
b) choose series of sessions that will provide training of profile;
c) from all sessions included in the chosen series to choose those that will be used for training.

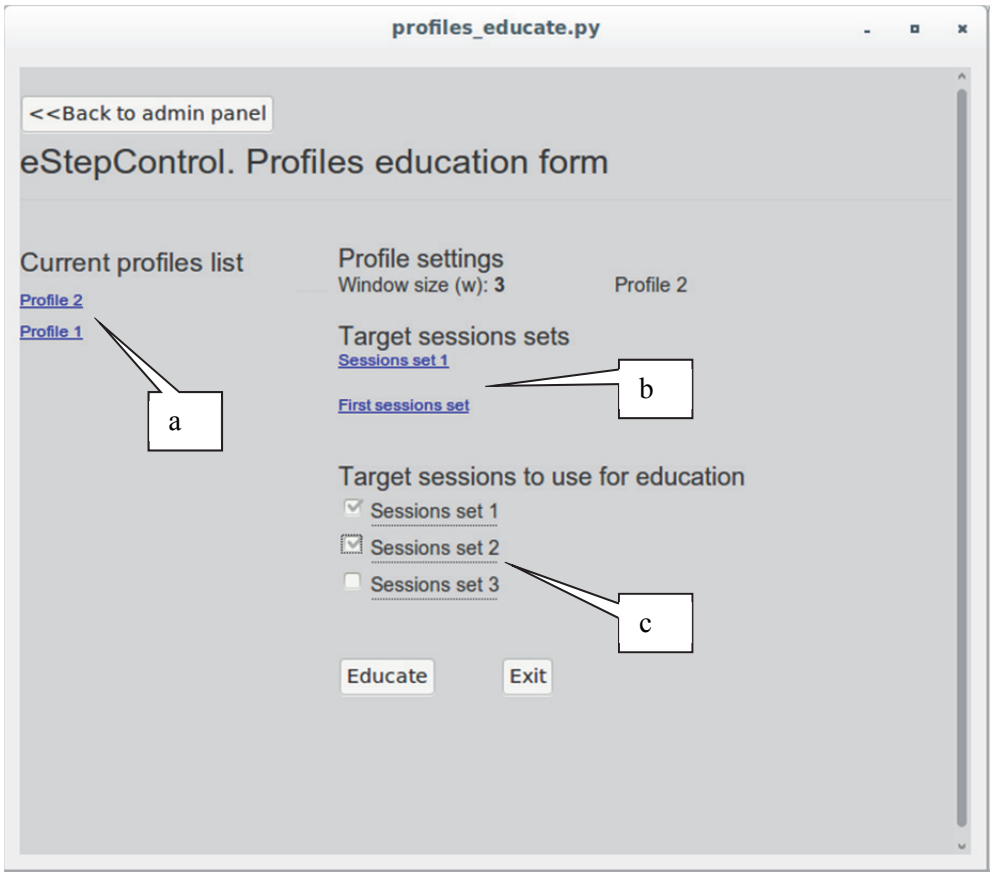


Fig. 4.3 Interface for learning profile based on new data on the behavior

4.2.3. Experiments management module

This module manages direct initialization of experiments. Every experiment has a purpose, base data for eash is set of used behavior profiles and involved set of sessions.

4.2.4. Experimental realization module

Main purpose of this module is experimental process management. After creation, the experiment can be carried out more than once, its data can be changed, and new sessions can be added, and internal parameters can be changed (Fig. 4.4).

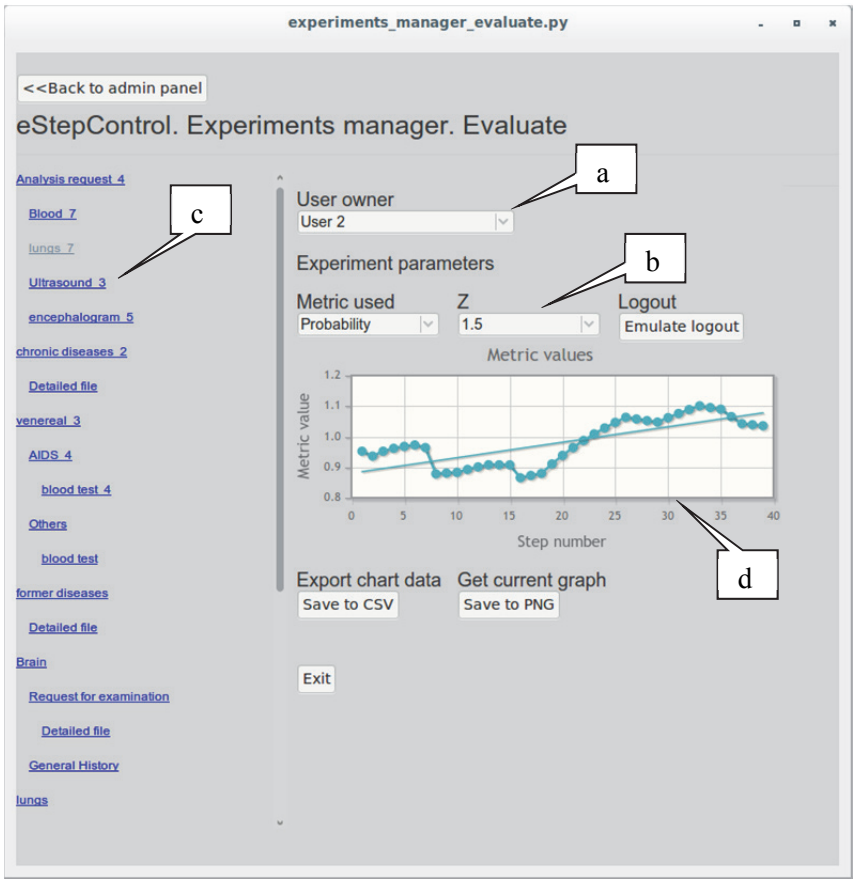


Fig. 4.4 Form of experiments evaluating in manual mode

During realization of experiments, using manual navigation mode following steps should be implemented:

a) to choose the target user, whose behavior is emulated;
b) to determine the anomaly calculation settings;
c) navigation by available categories; each action is considered an atomic query and its anomaly level is calculated;
d) to create a graph using all obtained values of the anomaly levels.

4.2.5. Result showing module

Typicaly, result of the experiment can be presented as dynamics of change of anomaly level. In this case, the module will show the graph. Specialized experiments have other

purposes − accordingly get other results. In this case the module can show the graph or to provide raw data for their processing in MS Excel [10] or Statistica [44]. All actions made at the level of graphic interface, at lower level Python also provide detailed information about the inquired made operations (Fig. 4.5).

```
Set node properties: {u'counter': 22}
Check node by name existing: 1|2|0|0|0^1|0|0|0|0^1|0|0|0|0
ans: True
Node not added, found existing with same name (1|2|0|0|0^1|0|0|0|0^1|0|0|0|0)
Recalculated properties for node: 1|2|0|0|0^1|0|0|0|0^1|0|0|0|0
Request for properties for node "1|2|0|0|0^1|0|0|0|0^1|0|0|0|0" returned: {u'co
nter': 10}
Properties setted:
{u'counter': 11}
Set node properties: {u'counter': 11}
Requested add arc between nodes: 36 and 37
Check arc between nodes: 36 and 37
Arc found: (36, 37)
set_ark_properties
ark_name: (36, 37)
properties_holder_obj: {u'counter': 11, u'transition_probability': '1.0'}
Outgoing arcs for node:
1|2|0|0|0^1|2|0|0|0^1|0|0|0|0
****11***
set_ark_properties
ark_name: (36, 37)
properties_holder_obj: {u'counter': 11, u'transition_probability': '1.0'}
TP: 1.0
Arc between 1|2|0|0|0^1|2|0|0|0^1|0|0|0|0 and 1|2|0|0|0^1|0|0|0|0^1|0|0|0|0
```

Fig. 4.5 Detailed log of profile training - high-level operation

4.3. Limitations of test subject domain

Subject domain has strict requirements to rate of processing of every transaction: the upper threshold of time of processing for 100 models is selected, which is equal to 500 milliseconds. In addition, there is the requirement to use typical server equipment without acquisition of expensive high-performance devices. Depending on processing power of equipment, rate of calculation for one model can vary. At present, the algorithm of calculation metrics and updating of model, realized in language Python, uses about 50 milliseconds of processor time on the computer Intel Core i3 560 + 4Gb RAM DDR3 under a management OS Ubuntu. However, it is necessary to take into consideration time consuming for initialization Python interpreter, time consuming for loading of model from a database, receipt of parameters and providing of result. Total time can achieve 200 and more milliseconds for one script; it does not allow sequentially process 90 models for 500 milliseconds. It is necessary to take into consideration that, in spite of the fact that physically a processor Intel

Core i3 560 has only two cores due to technology Hyper – Threading, each core can simultaneously execute two flows of operations transparently for the operating system.

There is the requirement in optimization of processing, usage of parallel calculations, involving all possible cores of processor, creation of connection pool with a database, usage of effective methods of long-term storage of models, as well as in effective distribution of resources for processing plenty of simultaneous queries.

Because of presence of these limitations, it was decided to use the experience and some programmatic tools applied for creation of highload-oriented electronic systems.

4.4. Basic descriptions of distributed model for realization the system

Practically all servers realized in language Python apply one of two approaches to handle incoming requests:

- usage of system processes (heavy servers) for input queries processing;
- usage of light-weight processes (lightweight servers) − system threads.

4.4.1. Initialization of system process for every query processing

In case of such method at the receipt of query, a server creates system process that processes a query regardless of server work. It allows effectively processing many simultaneous queries. However, creation of system process is quite resource intensive operation, and often the system has strict limitation on the amount of processes available to be used at the same time. Their amount rarely is more than 100, even on powerful modern servers with latest server operational systems installed. Mainly it is related to relatively large costs on planning of implementation of the operating system processes.

As a variant, to save time for initialization of process, a server can use pool created before; it increases efficiency, but does not eliminate the problem of limitation on the amount of available processes.

4.4.2. Usage of system threads for every query processing

Depending on realization of threads in used OS, the amount of resources for creation of thread can be approximately equal to the resources for creation of system process or less.

However in this case limitation is higher, and in case of large number of simultaneous connections (from thousand and higher) this model can be disabled because of the following reasons: expense of address space on a stack for every thread, large load on planner and limitation on total amount of threads in the system.

Obviously, that the second method allows processing simultaneous amount of connections within the framework of the given requirements (described in chapter 4.3); however, it has some defects not allowing to apply it in such systems. Basic problem is that with the increase of load, the requirements can change.

The server requires many resources for support of threads, but the main load however lies on the modules and it is necessary to take into account total load.

Because of these defects, there is the requirement to use the server resources more effectively for processing large amount of simultaneous queries.

4.4.3. Deferred approach

Deferred approach becomes more popular [92, 81]. The essence of this method is the following: at the receipt of query the module responsible for its processing is called; its function is to process event "*Calculation completed*" and then a server "*forgets*" about the received query and does not consume resources for its support. Later a query is fully processed, a server receives a signal about occurrence of event "*Calculation completed*" and a result of calculation, function that serves a response is called.

Conception of Deferred method differs from typical methods also from the point of view of the program realization. The function of processing target task using data from other servers is called from the code of server. As ideology of deferred method does not imply expense of resources on expectation of response, this function returns a result at once, in spite of it is not calculated. It is achieved by return of special object "*deferred*", to which necessary functions−listeners (at least, listeners of receipt a result and error) are added, and then practically all resources expended on processing query are unallocated. The following stage will be only at the moment of receipt result of implementation the inquired operation.

From programmical point of view, "*deferred*" object allow to return object, what provides ability to hook any asynchronoius events what will happen with returned data later, after object is received. This feature allow avoid situation called "*callback hell*", which makes structure of created programm more complex without actual needs for it.

Deferred is not a "*silver bullet*" allowing the server to process the number of simultaneous queries, expending even less resources than using the system threads. Its advantages are shown only in case, when a server does not engage in calculation of the inquired data and send them for processing to other services (network or local).

On Fig. 4.6 fundamental organization of processing queries is shown, using deferred method, but in case if a server processes data. Obviously, that there are no advantages in comparison with usual FIFO queries. Resources are additionally expended to maintain a context of listeners of every query, and logic of realization becomes more complicated.

However, in case if queries are processed by off−site services, efficiency increases significantly. On Fig. 4.7 it is shown that total time of processing already three queries can be significantly less.

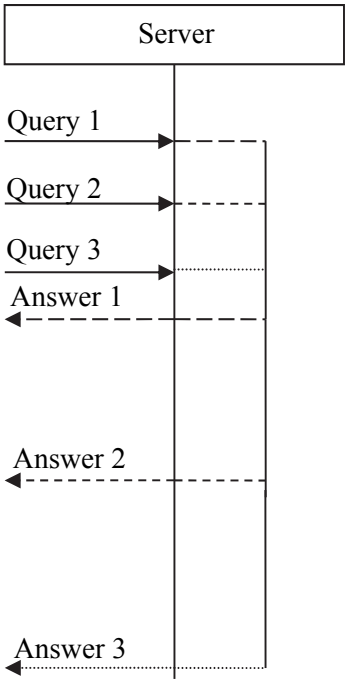


Fig. 4.6 Queries processing by server

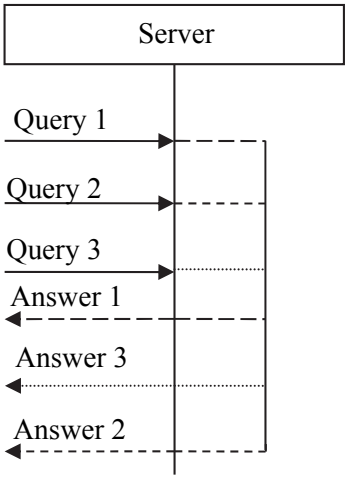


Fig. 4.7 *Deferred* using remote services

4.4.4. Dependence of efficiency from work mode of central programmatic server

Based on present requirements, usage of the server based on the call of system processes is not possible, so a selection has been made between usage of lightweight system threads and *Deferred* approach. As all tasks are processed on one processor, basic advantage of *Deferred* cannot be realized in full. However, for the ground of selection of the type of server the tests of both selected methods have been carried out.

4.4.5. Storage and access to data

Nowadays lot of electronic systems created specially to work under high−load uses non−relative databases named NoSQL [86] solutions. So data warehouse is realized not with usage of relational method, but exactly NoSQL decision is selected. Every model is identified on the unique key that allows avoiding the expenses of productivity, related to difficult SQL queries processing on selection/ updating data.

For storage of the models of user behavior represented as serialized objects, containing metadata models and graph containing a model, one of the popular systems of such type is used: NoSQL key/value data warehousing *Redis* [89].

REDIS is expansion of general idea of key-value, in which not only primitive operations of data receipt/ warehousing are supported, but also more difficult data structures are supported without significant decrease of efficiency: *binary-safe strings*, *hashcodes*, *lists*, *sets*, *sorted sets*.

Other important advantages of *Redis* is Sets, Lists with $O(1)$, push operation, "*lrange*" and "*ltrim*", server-side fast intersection between sets, are primitives that allow to model complex problems with a key value database.

4.4.6. Work with graph of the profile

For realization of basic operations with graphs, the library *NetworkX* is used [67], which is independent module for the language Python for work with graphs, trees, networks and other tree structures.

Possibilities of the library allow freely operating very large network structures of level of graph with 10 million nodes and 100 million arcs between them; it is too much, because the graphs of models will not achieve such volume of behavior. As the library is based on low-level structure of language Python data "dictionary – dictionaries", memory is expended effectively, the graphs are well scaled, depend on the features of operating system, in which a script is executed, not significantly, and comply with popular direction on the analysis of data from social networks and graphs [85].

4.4.7. Testing of query processing speed

As candidates for the role of basic server processing the incoming requests for the calculation of the anomaly level in the system three popular Python servers have been chosen: Twisted [29], Tornado [102] and Cyclone [20].

Only *Twisted* supports *Deferred* method. All three servers as default use system threads for processing input data.

Testing methodology is the following − the modified code of server is activated, in which emulation of processing difficult query for 0.005 seconds of processor time is added. Many parallel queries have been created for target server, and final statistics of their processing time has been collected.

Testing *Deferred* approach, delay in code is meaningless, but additional service is used − local server Apache available on other port, which also returns the inquired data in

0.005 seconds. Testing has been implemented, using cantilever program ApacheBench [3]. The following command was used:

```
ab -n 800 -c 100 http://localhost:8007/,
```

where: *ab* − a command calling listener of tests;

n − total amount of queries;

c − amount of simultaneous queries;

http://localhost:8007/ − the name of host and port for testing.

4.4.7.1. *Twisted* server in processing mode without deferred

Any (within the framework of the conducted experiments) amount of simultaneous queries has been calculated by this configuration without the change of time on processing one query. It shows that modern OS processes 100 threads created simultaneously without significant delay. On Fig. 4.9 a you can see a graph of dependence of amount of queries processed in a second for different amount of simultaneous connections to the server. On Fig. 4.8 and Fig. 4.9 time of all queries processing is shown for different amount of simultaneous connections to the server.

Concurrency Level	100	75	50	25
Time taken for tests	4.44	4.44	4.42	4.43
Requests per second	180.15	180.4	180.96	180.21
Time per request	555.10	415.73	276.30	138.72
Mean time per request	5.551	5.54	5.52	5.54

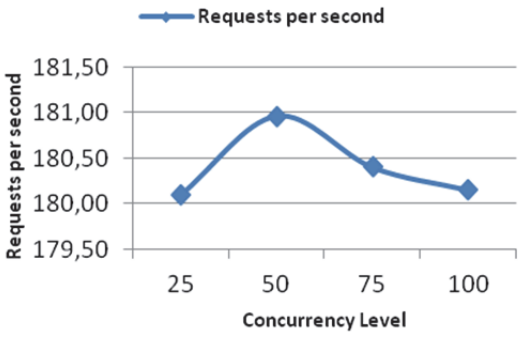


Fig. 4.8 *Twisted*, concurrency level VS requests per second

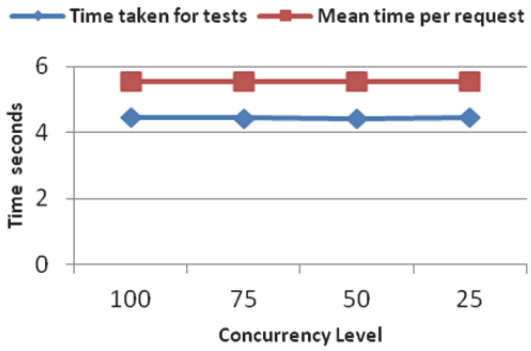


Fig. 4.9 *Twisted*, time taken for tests VS mean time per request without *Deferred*

4.4.7.2. *Twisted* server in deferred processing mode

By the increase of amount of simultaneous queries (Fig. 4.10, Fig. 4.11) mean time of processing every query increases, however if amount is less, the results are better than using threads.

Concurrency Level	100	75	50	25
Time taken for tests	4.23	3.78	3.18	3.04
Requests per second	189.09	211.28	251.07	263.10
Time per request	528.86	354.97	199.14	95.02
Mean time per request	5.28	4.73	3.98	3.80

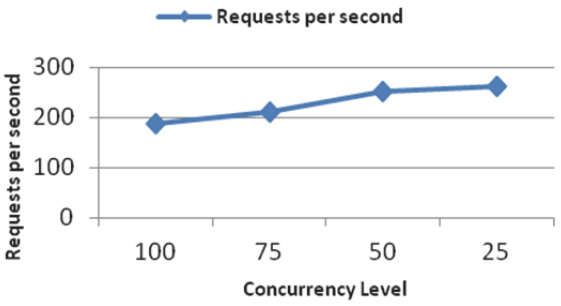


Fig. 4.10 *Twisted*, concurrency level VS requests per second with *Deferred*

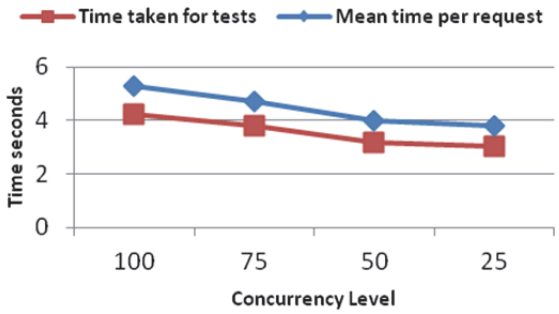


Fig. 4.11 *Twisted*, time taken for tests VS mean time per request with *Deferred*

4.4.7.3. *Tornado* server in processing mode without deferred

The result is similar with *Twisted* without usage of *Deferred*, only time used for processing of every query is rather more (Fig. 4.12, Fig. 4.13). It also shows effective use of *Twisted* system threads by the server. *Tornado* framework main fosus is to handle more than 10000 income connections (10K problem), so main response time may be worse comparing with *Twisted*.

Concurrency Level	100	75	50	25
Time taken for tests	4.47	4.49	4.49	4.50
Requests per second	178.77	177.85	177.93	177.76
Time per request	559.38	421.70	421.50	140.63
Mean time per request	5.59	5.62	5.62	5.625

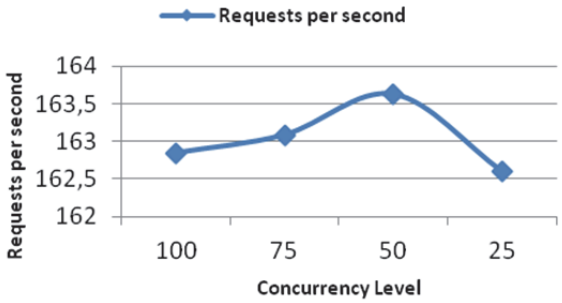


Fig. 4.12 *Tornado*, concurrency Level VS requests per second without *Deferred*

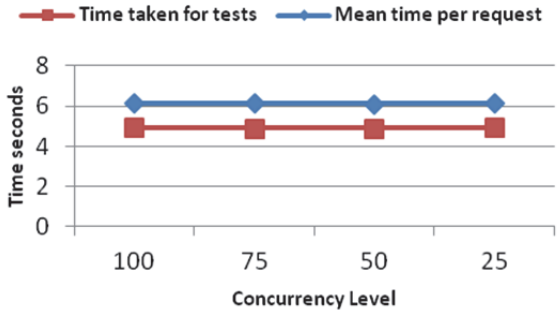


Fig. 4.13 *Tornado*, time taken for tests VS mean time per request without *Deferred*

4.4.7.4. *Cyclone* server in processing mode without *Deferred*

This server is created on the basis of *Twisted* protocol, so similar behavior is appropriate. The result is also predictable; it showed rather large expenses of time (Fig. 4.14, Fig. 4.15) for processing every query in comparison with *Twisted*.

Concurrency Level	100	75	50	25
Time taken for tests	4.91	4.90	4.88	4.92
Requests per second	162.85	163.09	163.63	162.60
Time per request	614.07	459.87	305.56	153.75
Mean time per request	6.14	6.13	6.11	6.15

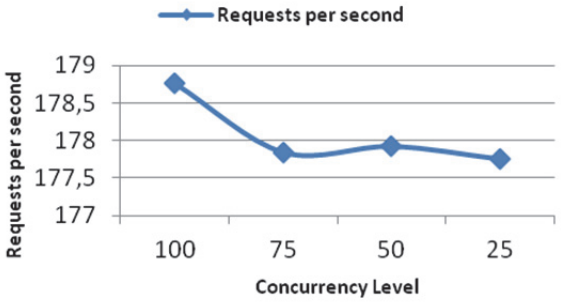


Fig. 4.14 *Cyclone*, concurrency Level VS requests per second without *Deferred*

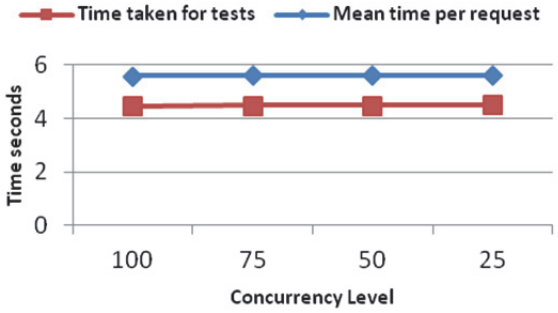


Fig. 4.15 *Cyclone*, time taken for tests VS mean time per request without *Deferred*

4.4.7.5. Estimate of performance of PHP server implementing a remote service

To estimate the impact of rate of processing every query at emulation of remote server using PHP, its testing has been implemented.

However, the server works quite well and increasing simultaneous queries to 100; mean time of processing every query did not change (Fig. 4.17, Fig. 4.16) practically.

Concurrency Level	100	75	50	25
Time taken for tests	0,47	0,47	0,46	0,43
Requests per second	1672	1701	1710	1853
Time per request	59,78	44,08	29,23	13,48
Mean time per request	0,59	0,58	0,58	0,53

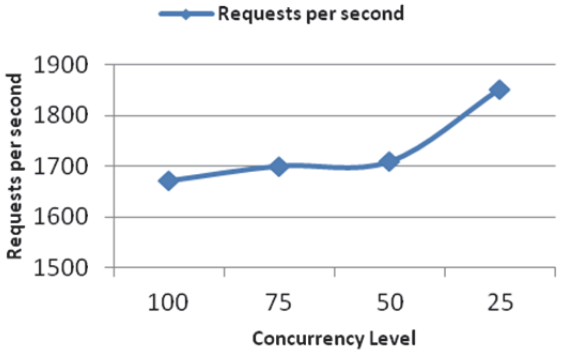


Fig. 4.17 Apache, concurrency Level VS requests per second

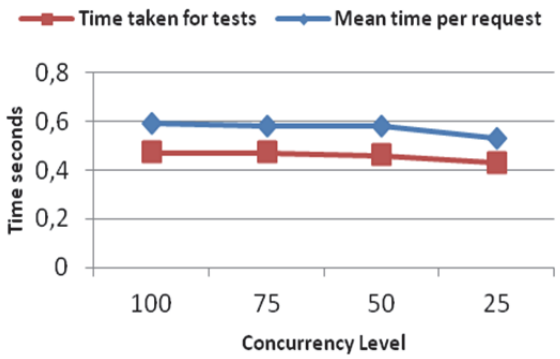


Fig. 4.16 Apache, time taken for tests VS mean time per request

4.4.7.6. Processing of test profiles

The results shown above can be used to estimate pure productivity of servers.

However to make the terms similar to real, estimation taking into account the operations of loading and storage of model is also important. To estimate these parameters, additional research has been executed.

At first, estimating of time on creation and storage of profiles in *Redis* directly from Python script without usage of servers has been implemented, because the generation of models will be executed separately from their processing by the servers.

Typical scheme of implementation similar process usually is linear model that is shown on Fig. 4.18 when iteration of the loop of random model executes all operations on creation and storage sequentially.

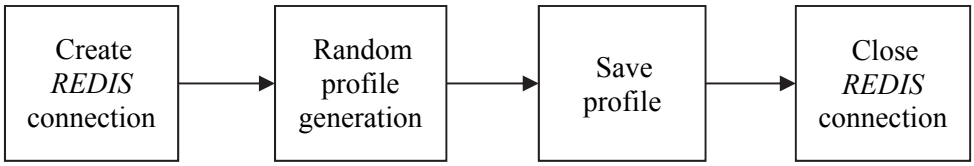


Fig. 4.18 Typical graph of model processing

As a code, this procedure can be shown as follows:

```
r = redis.Redis(host='localhost', port=6379, db=0)
graps_in_test_min = 1
graps_in_test_max = 150
for y in xrange( graps_in_test_min, graps_in_test_max ):
    for x in xrange(0,y):
        graph_key_name = "users_group_%s" % str(x)
        print graph_key_name+' generated'
        rnd_graph = gen_random_graph(50)
        data_string = pickle.dumps(rnd_graph)
        r.set(graph_key_name, data_string)
```

Additionally to linear logic of storage of models Redis allows to make package implementation of block of operations Fig. 4.19, that in theory must allow more effectively executing the block of operations in the loop, not consuming resources on initialization and closing of connection with Redis on every iteration.

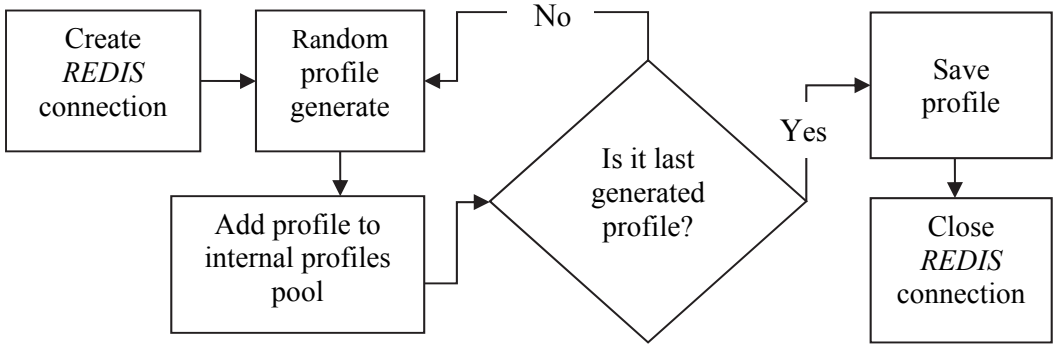


Fig. 4.19 Scheme of batch processing

As a code, this procedure can be shown as follows:

```
r = redis.Redis(host='localhost', port=6379, db=0)
graps_in_test_min = 1
graps_in_test_max = 150
for y in xrange(graps_in_test_min,graps_in_test_max):
    pipe = r.pipeline()
    for x in xrange(0,y):
    graph_key_name = 'users_group_'+str(x)
        print graph_key_name+' generated'
        rnd_graph = gen_random_graph(50)
        data_string = pickle.dumps(rnd_graph)
        pipe.set(graph_key_name, data_string)
pipe.execute()
```

On Fig. 4.20 time of creation and storage 10÷140 models having a random structure (but similar to real) is shown. Obviously, in the terms of specific of this task (namely as Redis server is on the same computer on that testing is executed, and all operations are made through main memory), usage of package processing of queries does not provide time saving.

Then estimating of load time of random model has been made also without usage of servers, based only on the mechanisms of optimization parallel calculations (Fig. 4.21) built-in the operating system.

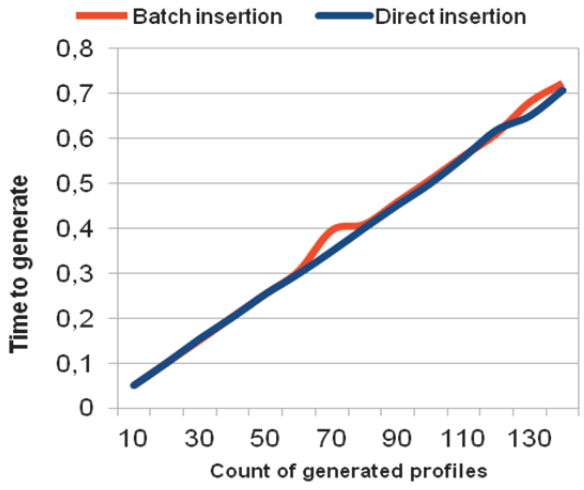


Fig. 4.20 Time for different count of profiles creating

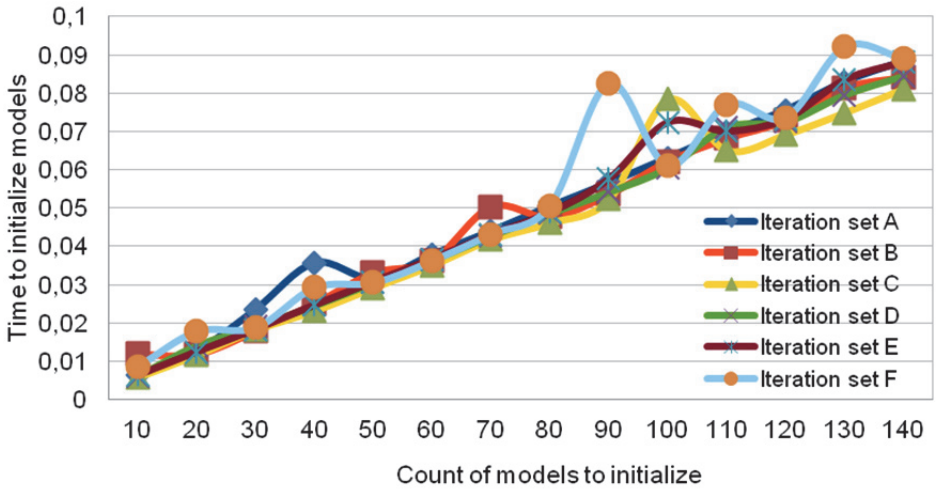


Fig. 4.21 Random models loading time

Sets of data A, B and C show three processes on loading 10÷140 models. Obviously, that total dynamics has a linear structure, and splashes have random character and depend on off-site processes in the system.

Sets of data D, E and F also show time consumed for loading 10÷140 models, but additionally after loading operations on the search of characteristics of tops and arcs are executed, which are typical for the used algorithm. Obviously, that in comparison with time of receipt the model, operations require little resources.

On Fig. 4.22 total mean time of response of all servers used in this testing is shown, but not with the emulated artificial delay, but with real loading of the model and implementation of several operations. Obviously, that, as well as in initial tests, *Deferred* approach shows the best results.

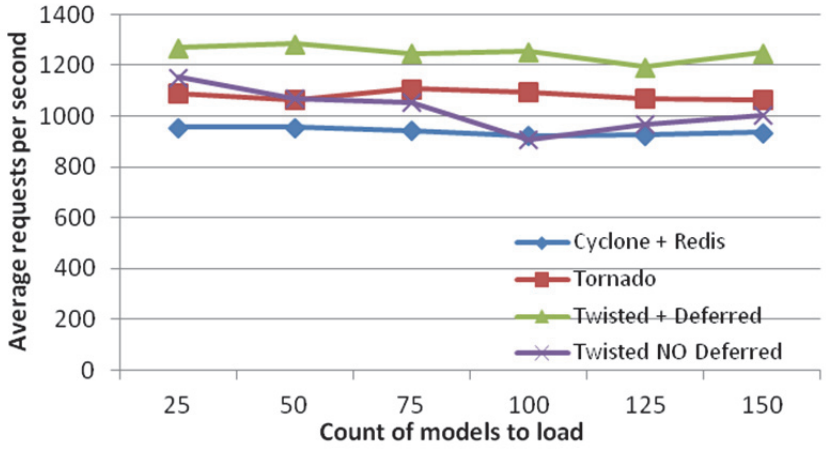


Fig. 4.22 Response time for different servers

4.5. Impact of hardware and software on quality of the program work

Nowadays many algorithms of data processing require large hardware resources for functioning. It can be explained by the increase of complexity of algorithms, as well as by volumes and structure of the processed information. Usual practice in such cases is the increase of amount and power of processing equipment, in particular, transition to cluster or cloudy calculations. However not always such method is justified financially, and there is a question about implementation such tasks using available equipment, but using different optimization methods. In particular, research of possibilities to decrease complexity of the used algorithms can provide the best result than usage of more productive equipment.

Efficiency of research environment. In the process of modern scientific research, many tools for data generation, processing, analysis and visualization are used. Usually for every task, specific software is used, which is oriented only to a certain range of tasks. However, increasing the amount of such programs, it is becomes more difficult to transfer data for the analysis between various instruments, each of them uses the appropriate format of input and output data. Significant feature is monitoring of the executed transformations in order to understand, how final results of the experiment depend on initial. As well as it is important to control the stages, at which an error may arise, in order to understand reasons of the error − human factor, error in input data or in the module of analysis.

Different instruments are realized on different programming languages: C [105], C++ [24], Fortran [38], Cobol [99], Perl [69], or as closed programmatic complexes, such as MatLab [37], MathCad [63], Mathematica [5], Statistica, SPSS [51]. Large number of various modules complicates the research process. Additional important lack is complexity or impossibility of adjusting direct co−operation between the modules in order to make data exchange (pipelines) without usage of intermediate files of results. As a result, significant part of paper has been devoted to solution of problems of data converting from one format to other.

The above mentioned not only complicates the research process, but also increases its duration, as well as increases probability of errors.

It is logically to suppose that the presence of single environment, in which an experimenter will be able to realize a lot of necessities will allow to achieve the best results,

because research resources will not be expended on data synchronization between different technologies.

One of the best variants to solve this problem is usage of language Python [59, 65], as well established and highly loaded in the creation of systems and to implement a variety of scientific problems.

By development of experimental system, Python has been used as a general research platform. Ii is interesting, that nowadays there is a recommendation also in various scientific articles to use exactly Python, because its syntax frequently is more expressive, describing difficult algorithms. At present, there are possibilities to realize any stage of project activity within the framework of single method and standardized types and layouts of data in this language. There are modules, allowing generating, analyzing, changing and visualizing data, and Python, being general language, is used for establishment of communication between these modules, as well as for management and analysis of data flows.

Some sources assert [87], that rate of implementation of the programs of Python (as the interpreted language) is 20−30% lower than using the programs on compatible languages, such as *C*. It can be applied to some Python programs, but the methods to increase rate of implementation the programs are already developed, for example, realization of critical areas as the modules in *C* language or usage of *Just-in-time compiling* of Python code [16]. As well as realization of difficult algorithms on the interpreted languages requires 50% less time than on compatible languages.

In turn, there are some problems, using Python language. As this language is the interpreted language with dynamic typification, there is a threat of errors in time of implementation. As one of variants to solve this problem, it is possible to decrease their impact by coverage of larger part of code by automatic tests.

As additionaly advantage of Python can be pulled availability of lot good supported extension libraries, what quite easy can be used by researcher even having small programming skills.

Based on analysis above Python language was selected to implement target system of this research.

4.6. Methodology of the created system introduction

In the terms of real tasks, not always it is possible to understand the efficiency of the developed approach within the framework of different target systems. Approaches for the construction of information systems can differ significantly depending on requirements to

their structure, level of availability and data safety. As a result, there are many **complex** information systems with different architectures, platforms and methods of realization, but it is important for all of them to get possibility to determine the cases of user anomalous behavior.

4.6.1. Requirements for the created system

How can the organizer of complex information system determine that approach for anomalous activity detection developed within the framework of this research can be used?

Basic descriptions of the target information system are specified below to show necessity and validity of introduction the system similar to developed in this thesis:

- Presence of sensitive data. If the system does not use such data, necessity to introduce such complex approach decreases significantly.
- Distributed structure of the target information system realized based on unified integration platform. If the target information system consists of dissimilar blocks not unified in common logical scheme, the possibilities of introduction the created system can be limited.
- Presence of possibility to store and access to data about user queries. If such data are not available, there is no basis for training of behavior profiles.
- Presence of safety providing module in the target information system. The developed approach allows estimating the anomaly level of user queries, but external safety module makes decision about the actions that have to be done.

4.6.2. Requirement features for introduction

Introduction of such functionality has to be available within the framework of set time, budget, software and hardware expenses. As a result, there is a requirement of unification of external interfaces of the created system in order its introduction in different target platforms will be limited with setting and testing of availability of the required software services.

As specified in the section 2.4, the created system for the analysis of the anomaly level of user query requires at least three types of the incoming data:

- unique identifier of user;
- identifier of his role;

- identifier of made transaction;
- + **additionally**, for training it is necessary to have data about the previous sessions of work with the system.

As a result, the system has to realize at least four internal interconnection interfaces for data transmission of each above−mentioned types.

Other interface that has to be realized is public network interface for the target system interconnection with anomalous activity detection module. It receives all requests for determination of the anomaly level and it returns the inquired data. General structure of such approach is shown on Fig. 4.23.

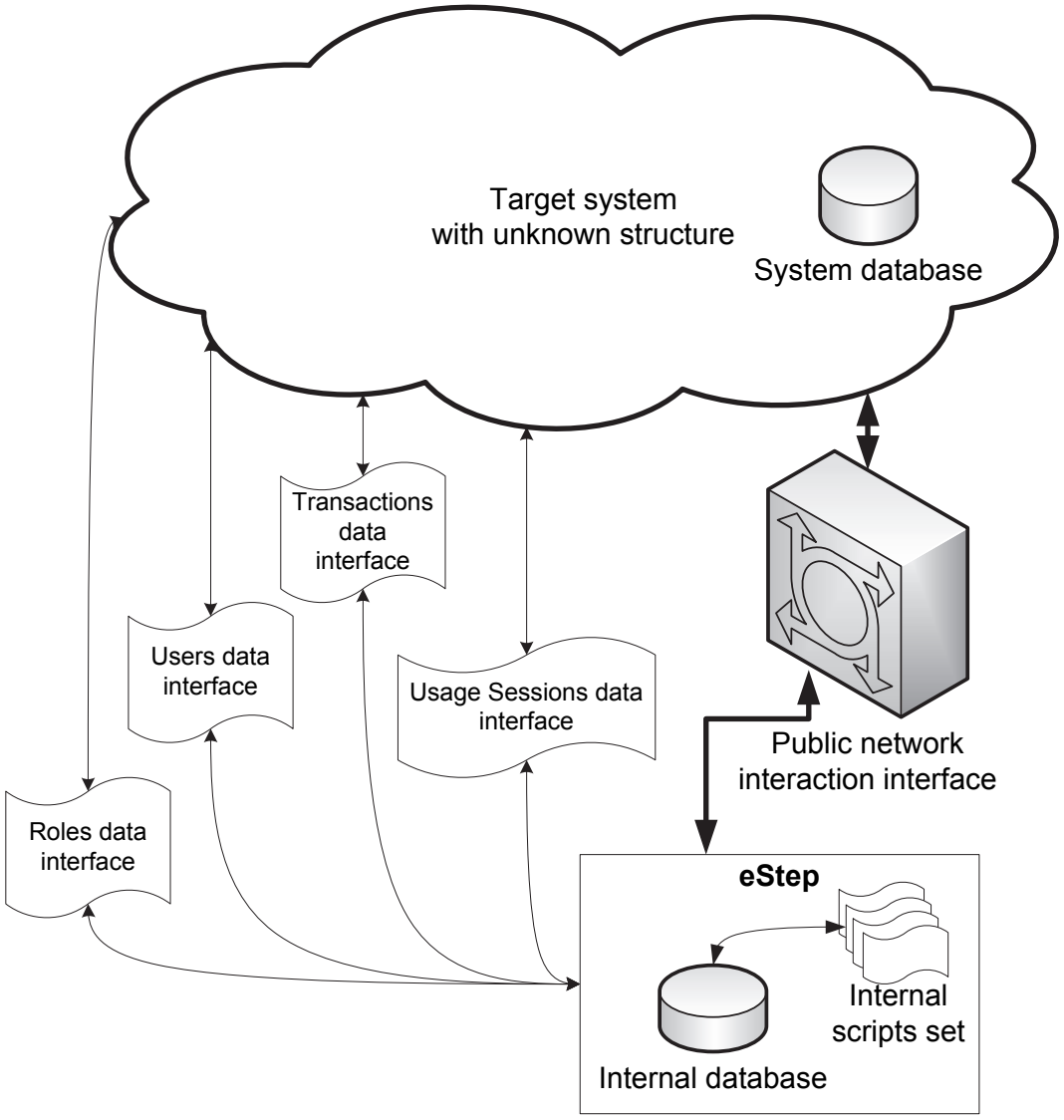


Fig. 4.23 The overall structure of interfaces interoperability

4.6.3. Types of availability of the target system data

Depending on the availability of all required data, it is possible to specify four types of the target systems:

1. ***full available*** − all required data are available;
2. ***partly available*** − the required data are available only partly;
3. ***unavailable*** − none of the types of required data is available directly;
4. ***specifically available*** − the target system does not provide the presence of all required software or hardware descriptions necessary for functioning of the system. As the example of such requirements the following list can be shown:
   - required software services:
     - programming language Python 2.7;
     - system library PyQT;
     - server Twisted set by Python;
     - Python library NetworkX;
     - availability of Redis service.
   - hardware possibilities:
     - availability of network interface of the target system;
     - at least one specified server for development of the system;
     - in case if the target system has distributed system, common point of connection has to be available that unifies all services for providing information to the system.

It is not possible to determine availability of required services of 1−3 groups automatically, it can be done only using expert analysis approach.

Availability of required resources of the fourth type can be realized in automatic mode, for this purpose the analysis of software and hardware possibilities of the target platform is used. Common presence and libraries versions what available in host system can easily be accessed and analyzed by setup script what can help automatize minimal software requirements checking by administration personel.

4.6.4. Variants of the system organizational structure

Depending on the type of the target system, described in the section, interconnection methods will differ. Detailed description of possible types presented in Table 4.1.

Table 4.1

Structure of public interfaces depending on the type of the target system

Type of the target system structure	***Architecture of public interfaces of the system***
Fully available	To use standard interfaces.
Partly available	For available data to use standard interfaces, for unavailable data realization of specific, non−standard interfaces is required. Aggregating of required data from several sources of the target system is used, or emulation of missing data on the basis of logic maximally suitable for the current target system.
Unavailable	All interfaces have to be realized as non−standard scripting interfaces.
Specifically available	In case of unavailability of software or hardware resources, the target system administrator will be informed. Then the following actions are possible: • termination of the system work; • attempt to set missing components automatically; • software emulation of the missing nodes; • use of replaced libraries for required software components (for example, use of other NoSQL archive in case of unavailability of Redis).

4.6.5. System settings

After the system is installed, it is necessary to set its internal parameters (see section 4.6.3). Quality criteria are the following two basic descriptions:

- **Classification efficiency** – the system efficiency level of detection anomalous behavior at the current values of internal parameters. In this case, the efficiency is correlation between correctly detected anomalies and the number of errors of the first and second kind [96].
- **Performance** – time for calculation of the anomaly of one query. The system has to provide rapid activity within the framework of the existing functional requirements.

Setting of internal parameters can be done both in manual and automatic mode. Verification can be made using the test set of histories of one user behavior with specified sets of *"normal"* and *"anomalous"* behavior. The test set is divided into two parts – *training* and *testing* selections.

- *Automatic mode* − in the cycle selects all possible variants of settings for each of parameters. Limit values for each variant are set by an expert. For each tested variant, the percent of correctly classified data is stored. As a result, the best variants of settings are sorted by quality level of data classification, exactly within the framework of specificity of the current target system.
- *Manual mode* − in this case an expert sets the values of internal parameters, starts verification on the basis of the test set and interprets the results. Such method cannot specify the most appropriate set of parameters, but can require less time for realization of all experiments in comparison with automatic mode.

Based on the results of made settings, the system will be able to start operation with high efficiency. Additional settings will not be required in future, because additional training will take place automatically in regular mode.

4.7. Summary

In basis for development of experimental system, it is necessary to describe the most important features of the target subject domain in detail, to take into consideration the limitations imposed on available resources. For providing of development of effective experimental system its structure is offered, the choice of technologies and the best software instruments for its realization is grounded.

In section 4.4, the process of choice of approach for realization of network interconnection in the developed system is described. Three possible approaches are considered: network flows, network threads and *Deferred*. *Deferred* has been chosen − as the most effective within the framework of the solved task.

Approaches used for storage and presentation of behavior profile objects have been grounded and chosen.

The set of experiments has been conducted to compare efficiency of considered approaches. As well as measurement of inquiry, processing speed has been done using different web servers: *Tornado*, *Cyclone* and *Twisted*.

The results of considered set of experiments show that the use of *Twisted* web server working in *Deferred* mode and storage of User Behavior Profile objects in NoSQL – Redis database, is the most effective from the point of view of access and User Behavior Profile processing speed.

- Complete description of all features of subject domain allowed creating experimental system that complies with them best of all.
- The detailed planning of the structure of created system provided practical basis for creation of the system that allows carrying out various experiments with UBP.
- Experimental testing of different variants of co-operation between servers allowed to select and to use the most effective method.
- Usage of language Python at all stages of creation of experimental system provided possibility to decrease time expenses for realization of the system.
- Determination of necessity criteria for introduction of developed system provided the possibility easily to specify, whether it is necessary to introduce it in the terms of specific of different target information systems.
- Differentiation of data availability levels in the target information systems allowed to describe the sets of operations required for introduction of developed system at each level.
- Description of possibilities for initial settings at the stage of introduction of the system allowed specifying important descriptions of efficiency of its introduction.

5. EXPERIMENTS WITH USER BEHAVIOR PROFILE

Experiments are the main tool of this research. Each set of the conducted experiments considers important part of descriptions. The first part of experiments is conducted in order to show fundamental possibility of the used approach within the framework of the set task. The aim of the second part of experiments is to assess dependence of accuracy of the anomaly level detection depending on the properties of these profiles used for training, as well as values of their internal parameters. General task of all conducted experiments is to complete comparative description of General User Behavior Profile and Personal Adaptive Profile of User Behavior in order to show advantages of the second approach.

As well as significant part of this section is devoted to description of the method developed for the generation of artificial data about user behavior having different sets of priorities in the target system.

5.1. The first series of experiments

At first, for conducting of experiments, programmatic realization of version *1.0* has been created. Its basic task was a fundamental estimation of possibility to use the base algorithm for detection of anomalous behavior. In the first version possibilities on collection and storage of data test sets, construction of UBP, as well as generation of artificial sets of transactions and calculation of their anomaly level, have been realized. It is worth noting that the first version of the system realized basic approach only, but the concept of adaptive profile has not been considered at this stage.

As basic programming language of the first version of experimental system, the PHP language has been used. The choice of PHP can be explained as follows: target information system functions on the Internet, and at present time PHP is a popular programming language for WEB. Database MySQL [60] has been chosen because of its popularity and wide possibilities. Both are open sources, both provides good data processing, functionality and stability. For presentation of graph, Graph Markup language GraphML [13] is used, based on XML structure and flexible in expansions. Being universal language for description of graphs, GraphML allows setting required complexity of the structures.

The model is based on the idea about General system of medical information about a patient [43], i.e., Electronic Health Record (EHR) as a set of services (reports) that allow obtaining different information about a patient.

Depending on specialization of doctor (user), he may be interested in one or other part of the system (different sets of services). For modeling of services of the system, general structure is created, part of which is shown as taxonomy on Fig. 5.1. Hierarchical models can be shown in such way, as well as in future analyzing the model it will be possible to take into account semantic distance between the inquired services.

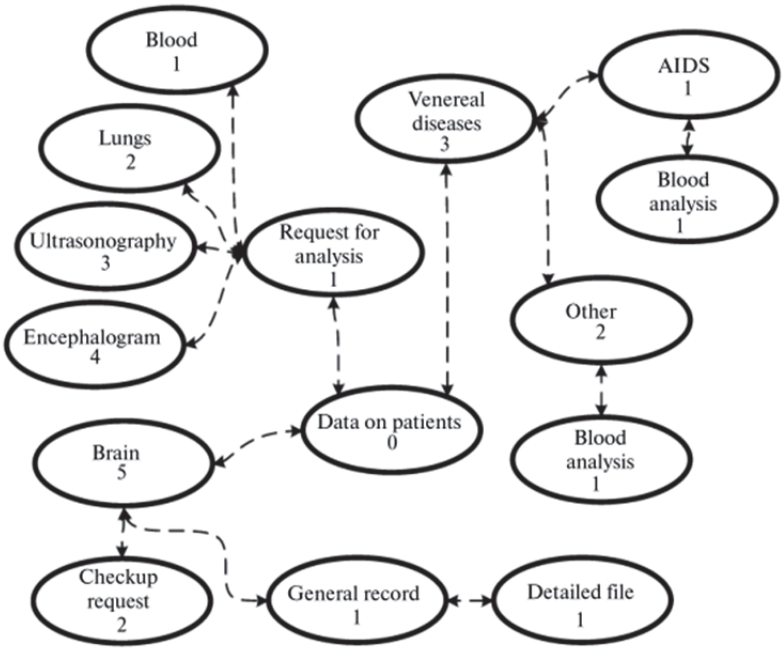


Fig. 5.1 Taxonomy of services interconnection structure

Information about sexually transmitted diseases and AIDS is included in a separate group as an example of presence in similar systems sensitive information and level of its importance.

Fig. 5.1 shows that the model is not uniform; there are services with general information, as well as information sources about the types of illnesses and organs. So, the attempt to model complex, heterogeneous environment of EHR is implemented.

Every layer has its own number that allows one to identify the inquired data according to transaction code. For example, a code (5,1,1,0,0) specifies query of file with detailed

information about brain state of a patient. Additional zero is added for normalization of key length within the framework of limitation complexity of used artificial data.

Within the framework of the first version of experimental program, only one type of metrics of estimation of transactions has been realized − *Probabilistic metrics*.

As estimation of efficiency of this method, it is planned to use the method of expert estimation [103] for verification the most probable situations, in case they are not presented in initial data used for training of profile. It is also important to engage experts in transition from artificial data of test task to real statistics.

To estimate classification abilities of the model, the following method has been used:

- Target type of hypothetical user is specified. A profile is created in visual mode according to the developed plan of the experiment, allowing covering different variants of queries that show interests of this user during a few work sessions.
- At the second stage, a profile is used in the mode of analysis, and the value of metrics for a current step, as well as graph of dynamics of its change for a current session, is specified. At this stage the system can be in three states:
  - ***Initial state* (*warming up*)** − At this stage information in suite of user is not enough for its effective analysis, and the values of metrics can differ significantly from small to high values unpredictable;
  - ***Normal behavior*** − At this stage the value of metrics has to be in stationary mode;
  - ***Anomalous behavior*** − At this stage the value of metrics has to increase continually.

The possibility of mixed behavior, when normal queries are mixed with anomalous, but anomalous are mixed with normal, has not been considered in this set of experiments.

5.1.1. Formation of initial terms

At this stage, it is necessary to form UBP describing any area of interests of simulated user. Two areas of interests are selected: **Brain** and **Lungs**. The complete set of the target types of queries is shown in Table 5.1.

In this case for full coverage of selected areas of interests, test sets of suites have to include all variants of crossing queries for both target areas: **Brain*** ∪ **Lungs** (a symbol * means that all subset of identifiers in area is used).

Table 5.1

Data for training

Area title	*Parental area*	*Key value*
Brain		5|0|0|0|0|
Query for investigation	Brain	5|1|0|0|0|
Detailed file	Query for examination	5|1|1|0|0|
Anamnesis	Brain	5|2|0|0|0|
Lungs		6|0|0|0|0|
Query for examination	Lungs	6|1|0|0|0|
Detailed file	Query for examination	6|1|1|0|0|
Anamnesis	Lungs	6|2|0|0|0|

To determine impact of constant parameters of metrics on its value, different values for a size of window w and constant Z are used.

All experiments are conducted in the following order:

- sequential queries to random category/subcategory from a set corresponding to current stage of the experiment: *warming up*;
- normal behavior;
- anomalous behavior.

The number of steps at every stage by default is equal to 50, unless special terms of the experiment are specified.

5.1.2. Results of experiments

Each exoperiments set used different values of internal parameters to find out his influence on final result of anomality detection. The results of experiments for different values of internal parameters of algorithm are shown below.

Checked only minimal and mostly possible values of parameters used. Bigger values, what theoretically may be used, but not have significant meaning, like window sizes bigger than 50, was not checked in this experiments set.

5.1.2.1. Experiment series № 1

For for first experiments set was chosen minimal interesting values of parameters to check. Exact values presented in Table 5.10. On Fig. 5.2 chart of anomality metrics level is presented.

Table 5.2

Settings of experiments set № 1

Window size	2
Value of parameter Z	1,5
Normal behavior	The value of metrics has been stabilized at the point of 3.4.
Anomalous behavior	The value of metrics started to increase constantly from mean value 3.3 at step No. 170 to 3.7 at step No. 220.

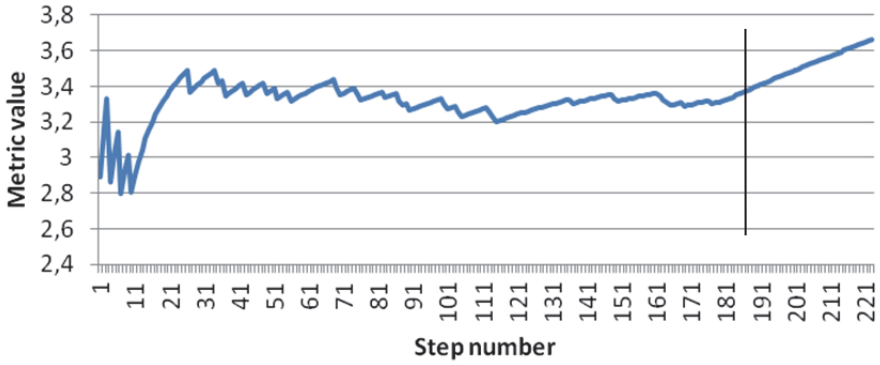


Fig. 5.2 Metrics behavior at w=2 and Z=1,5

5.1.2.2. Experiment series № 2

Window size was increased, exact values presented on Table 5.3, metric on Fig. 5.3.

Table 5.3

Settings of experiments set № 2

Window size	3
Value of parameter Z	1,5
Normal behavior	The value of metrics has been stabilized at the point of 3.5.
Anomalous behavior	The value of metrics started to increase constantly from mean value 3.4 at step № 160 to 1.36 at step № 200.

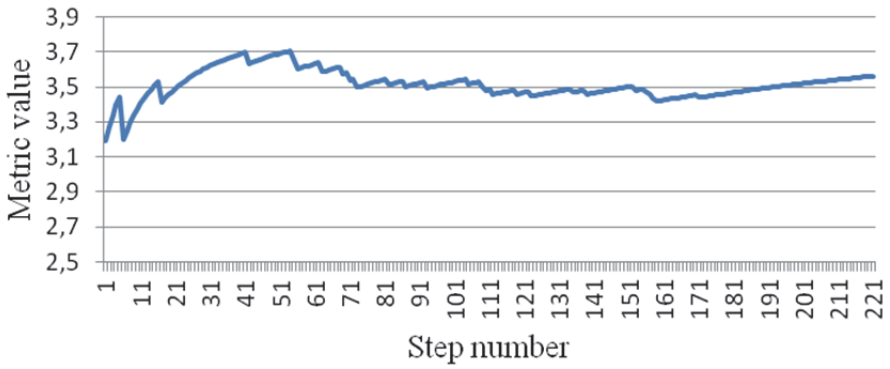


Fig. 5.3 Metrics at w = 3 and Z = 1,5

5.1.2.3. Experiment series № 3

Third series of experiments again have increased values of base parameters used, exact values presented on Table 5.4, dynamic of anomality metric changes on Fig. 5.4.

Table 5.4

Settings of experiments set № 3

Window size	3
Value of parameter Z	3
Normal behavior	The value of metrics has not very stable between levels from 3.3 to 3.6.
Anomalous behavior	The value of metrics started to increase constantly faster than with lower value Z, from 3.4 at step № 150 to 3.5 at step № 220.

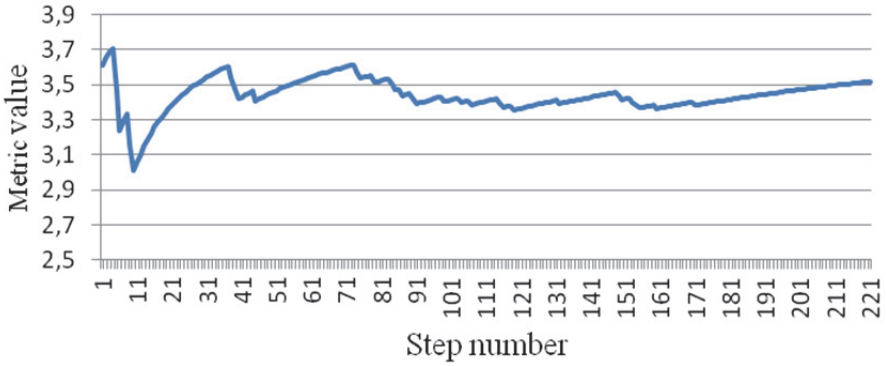


Fig. 5.4 Metrics at w = 3 and Z = 3

5.1.2.4. Experiment series № 4

This series of experiments again have increased values of base parameters used, exact values presented on Table 5.5, metric on Fig. 5.5.

Table 5.5

Settings of experiments set № 4

Window size	3
Value of parameter Z	4
Normal behavior	The value of metrics has not stabilized to step № 150.
Anomalous behavior	The value of metrics started to increase constantly from 3.30 at step № 150 to 3.5 at step № 220.

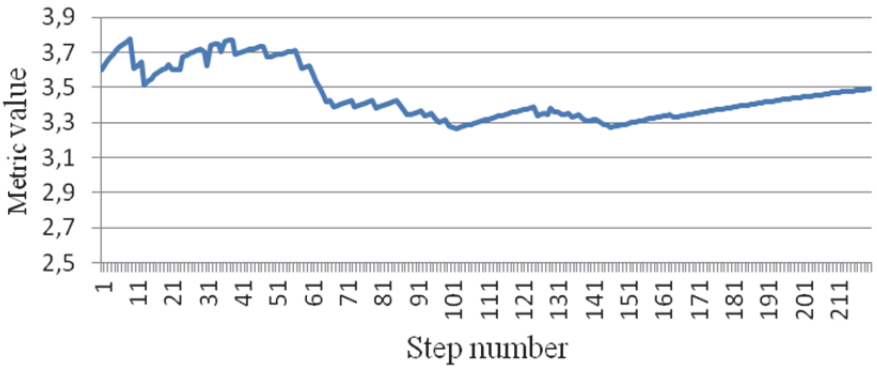


Fig. 5.5 Metrics at w = 3 and Z = 4

5.1.2.5. Experiment series № 5

Fifth series of experiments again have increased values of base parameters used, exact values presented on Table 5.6, metric on Fig. 5.6.

Table 5.6

Settings of experiments set № 5

Window size	3
Value of parameter Z	5
Normal behavior	The value of metrics has not stabilized to step No. 50, but has decreased constantly.
Anomalous behavior	The value of metrics started to increase constantly from 3.53 at step № 150 to 3.5 at step № 220.

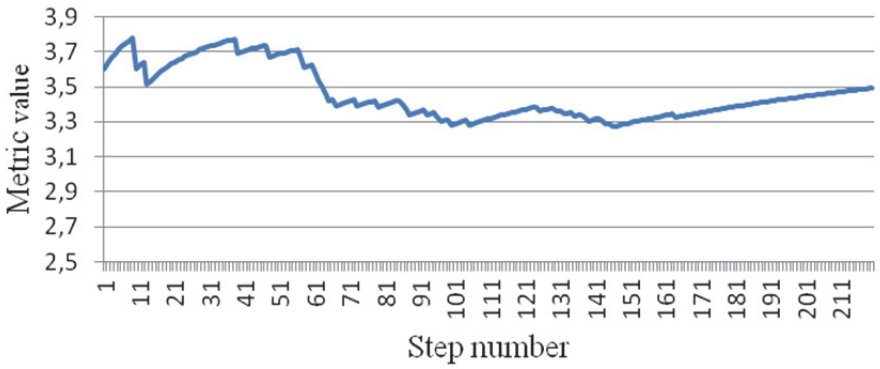


Fig. 5.6 Metrics at w = 3 and Z = 5

5.1.2.6. Experiment series № 6

This series of experiments again have increased values of base parameters used, exact values presented on Table 5.7, metric on Fig. 5.7.

Table 5.7

Settings of experiments set № 6

Window size	4
Value of parameter *Z*	5
Normal behavior	The value of metrics has not stabilized to step No.50, so we decided to continue testing within the framework of normal queries to 100 steps. As a result, increase of metrics stopped at level ≈ 3.3.
Anomalous behavior	The value of metrics started to increase constantly from 3.3 at step № 150 to 3.5 at step № 220.

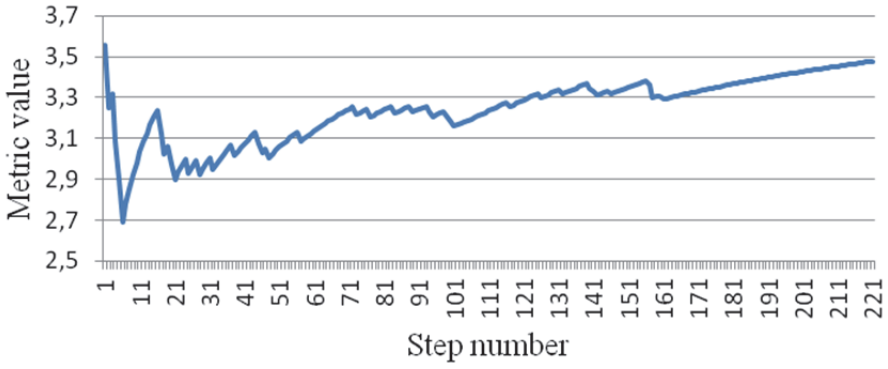


Fig. 5.7 Metrics at w = 4 amd Z = 5

5.1.2.7. Experiment series № 7

Seventh series of experiments again have increased values of base parameters used, exact values presented on Table 5.8, metric on Fig. 5.8.

Table 5.8

Settings of experiments set № 7

Window size	5
Value of parameter *Z*	5
Normal behavior	A dynamics is similar to the situation $w = 4$, $Z = 5$, when stationary mode at 50 steps has not been achieved. In addition, the amount of normal behavior queries has been increased to 100, as a result, metrics has been stabilized at level around 1.1. As well as it is shown that size of area "*warming-up*" of model increased, that corresponds to an increase of size of window and the amount of initial empty transactions.
Anomalous behavior	The value of metrics started to increase constantly from 1.05 at step № 200 to 1.15 at step № 270.

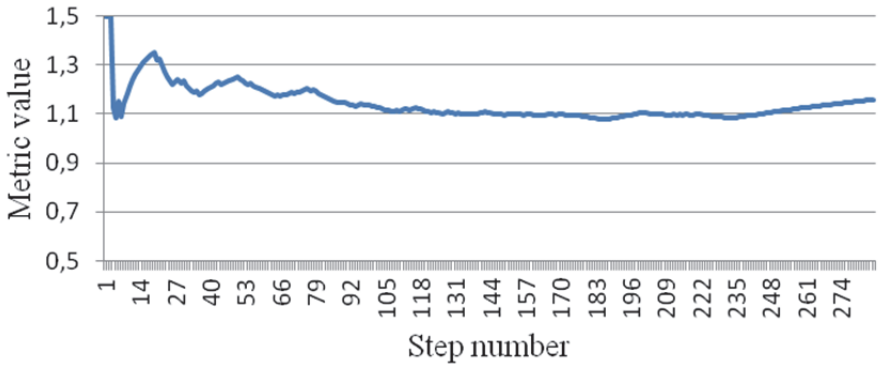


Fig. 5.8 Metrics at w = 5 and Z = 5

5.1.2.8. Experiment series № 8

Following series of experiments again have increased values of base parameters used, exact values presented on Table 5.9, metric on Fig. 5.9.

Table 5.9

Settings of experiments set № 8

Window size	6
Value of parameter *Z*	6
Normal behavior, Anomalous behavior	For this size of window and value Z, the dynamics of change of metrics become different (Fig. 5.9). Obviously, the amount of data during training is not sufficient; as a result, the existing arcs between nodes containing 6 identifiers are detected rarely. The graph shows that the existing arcs have been detected only in a few cases; in other cases, the metrics increased both in the mode of *"normal"* and *"anomalous"* use.

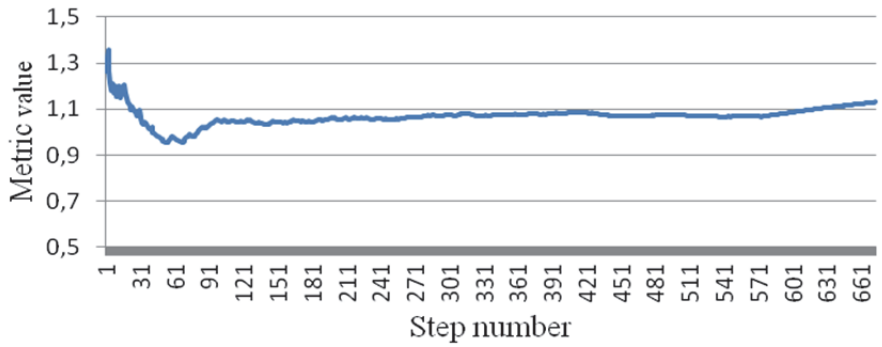


Fig. 5.9 Metrics at w = 6 и Z = 6

5.1.3. Statistical properties of profile – the experiment

It is important to show that the model created on the basis of certain behavior of user, regardless of input data, correctly divides normal and anomalous behavior and depends only on data used for its training.

Additional system, which allows generating many sessions on the basis of selections only from specified subset of categories, has been created.

At the stage of data preparing for the experiment, the amount of created sessions and steps in every session is specified, and to introduce additional element of randomness their amount each time is calculated using the following formula:

$$round(c-(c/d)) > c > round(c+(c/d));$$

where c – a size of created session;

i – desired amount of steps in one session;

k – the amount of created sessions;

d – variation in relation to size of session;

$round()$ − function that returns integer of fractional number expressed in round number.

Finally value of $c = round(i/k)$.

Desired value of parameters W and Z for created model is also set. Then according to generated set of sessions, we create a model that is stored in database for conducting experiments.

Stage of experiment realization consists of four following phases, what covers all steps of typical system usage in real life conditions:

1. Selection of model and determination of experiment settings. The amount of iterations for other phases of experiment is set.
2. Then three phases of generation of random steps from the subset of total amount of available categories follow. In each phase, only corresponding subset is used. Selection is equally possible for any key of target subset. Random key is selected for iterations and added to current set of keys of session, and then the value of metrics is calculated, which will be stored in history of experiment.
3. ***Warming up*** initial stage (balancing) of the model. Categories are selected from a set of keys corresponding to "*normal behavior*" of user. In this phase, information for correct work of algorithm is not enough, and results of metrics can be changed randomly.
4. ***Normal behavior*** – behavior is basic phase. Categories of set of keys of "*normal behavior*" are selected too. In this phase, the value of metrics has to obtain property of stationarity, because keys used for creation of model are analyzed.
5. ***Anomalous behavior*** – is final phase. At iterations, the keys are selected from set of "*anomalous behavior*". Cumulative value of metrics has to increase droningly.

5.1.4. Statistical properties of profile – the result of experiment

The results showed similar behavior of the model that does not depend on input data, on the condition that they are selected from a set corresponding to each phase of the experiment.

On Fig. 5.10 graphs of three experiments are shown simultaneously. Obviously, that general dynamics of change of value of metrics is similar, in spite of the fact that for every experiment unique trace has been created.

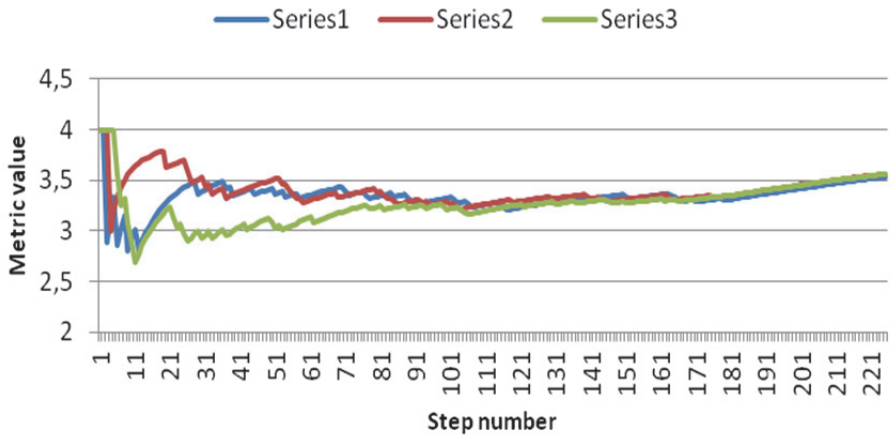


Fig. 5.10 Dynamics of values change for several experiments at the same time

5.1.5. Conclusions on the first series of experiments

As has been expected, the dynamics of changes in values of metrics changes at different sizes of window and other internal parameters of algorithm. In addition, we confirm the assumption that regardless of the appropriate inquired transactions, if they are inquired within the framework of the same user behavior, the values of metrics for all such sessions will have similar statistical descriptions.

5.2. The second series of experiments

The purpose of this chapter is to show formally that PUUBP will be more effective (in detection anomalous activity) than GUBP. In chapter 6, general theoretical methods on estimation of efficiency of similar information systems are considered; tools of behavior emulation are created. In this chapter, the author describes the set of experiments conducted for comparison of efficiency of the system using GUBP and PAUBP.

To answer determined questions, experiments have been conducted. It was decided to generate sets of vectors of transactions for training in automatic mode. The purpose of the first set of experiments was to process the method of generation of data about certain user behavior, using priorities of interests.

It is also important to compare characteristics of the profiles trained according to different selections.

5.2.1. Frequency characteristics of data about user behavior and their generation

For authentication of interests of certain class of users from general set of various types of queries, their sub-group has been determined, which describes a range of professional interests – *role*. Every type of query has the unique code – *transaction type identifier*. The obtained set of role transactions identifiers is shown in Table 5.10.

Table 5.10

Subset of interests of target role representatives

Interest title	***Transaction identifier***
Query of analysis	1|0|0|0|0|
Blood	1|1|0|0|0|
Lungs	1|2|0|0|0|
Ultrasound examination	1|3|0|0|0|
Encephalogram	1|4|0|0|0|
Chronic diseases	2|0|0|0|0|
Detailed file	2|1|0|0|0|
Sexually transmitted diseases	3|0|0|0|0|
AIDS	3|1|0|0|0|
Blood test	3|1|1|0|0|
Other	3|2|0|0|0|
Blood test	3|2|1|0|0|
List of old diseases	4|0|0|0|0|
Detailed file	4|1|0|0|0|

5.2.1.1. The first selection – the sets of behavior for GUBP

Average behavior is based on plenty of identifiers from general set of interests of target group of users. In general case there is equal probability of transition between all possible states.

For generation of such set of selections, a set that includes 100 sessions has been created. Each session consists of vectors (50 ± 5 identifiers of transactions selected randomly from general set). Thus, probability of selection of one or other transaction for all was equal. From a base set of interests (Table 5.11 − A) with equal probability 4987 identifiers of transactions have been selected. Data processing has been made using the module Data Analysis MS Excel [8]. To receive the amount of entry of every identifier, the following formula (a column with identifiers has index B) has been used:

`=COUNTIF(B:B; B6)` − for each of 4987 cells.

As a result, in a column C for every element, the amount of its entry to current selection is shown (Table 5.11 − B).

Table 5.11

Frequency table of the amount of transactions for GUBP

A) Example of base set of interests	*B) Amount of interests entry*
3|1|0|0|0| 3|0|0|0|0| 1|3|0|0|0| …	3|1|1|0|0| 367 3|1|0|0|0| 371 3|0|0|0|0| 389 …

Using possibility *Paste Special* → *Paste Values*, the formulas for calculation the amount are replaced by his values.

Then using Excel built−in features, copies have been deleted, and array of unique values has been formed as a result. As well as sorting on frequency in the reverse sequence has been done (least frequently used elements are the first). As a result, selection has been created; basic properties are shown in Table. 5.12.

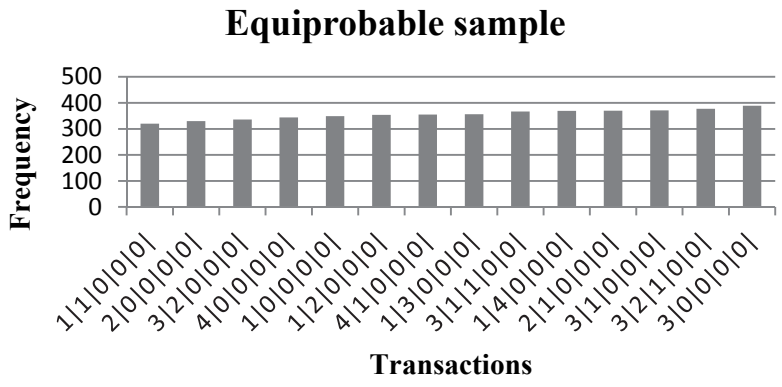
5.2.1.2. The second selection − the sets of behavior for training of personal UBP

This selection is a simulation of interests of one user. As in this case there is no necessity to average values of plenty of users, properties (structure) of this selection will differ from previous structure.

Let`s assume that a certain representative of researched role, as well as specialty A′ doctor, also is interested in the same set of possible queries. However, the specifics of his interests A″ can consist of more or less frequent usage of certain queries.

Table 5.12

Chart of GUBP frequency table

Frequencies of transactions presentation in created data set	*Descriptive statistics*	
	Mean	356
	Standard Error	5,14
	Median	355,5
	Mode	#N/A
	Standard Deviation	19,23
	Sample Variance	370
	Kurtosis	-0,39
	Skewness	-0,28
	Range	69
	Count	14

In this set of experiments generation of sets of identifiers of personal interest is based on adding of priorities to all possible types of transactions. Final data used for receipt of training selection are shown in Table 5.13.

Table 5.13

Base data for training of PERSONAL UBP

Interest title	Transaction identifier	Priority
Query of analysis	1|0|0|0|0|	1
Blood	1|1|0|0|0|	2
Lungs	1|2|0|0|0|	
Ultrasound examination	1|3|0|0|0|	
Encephalogram	1|4|0|0|0|	3
Chronic diseases	2|0|0|0|0|	11
Detailed file	2|1|0|0|0|	12
Sexually transmitted diseases	3|0|0|0|0|	4
AIDS	3|1|0|0|0|	5
Blood test	3|1|1|0|0|	6
Other	3|2|0|0|0|	7
Blood test	3|2|1|0|0|	8
List of old diseases	4|0|0|0|0|	9
Detailed file	4|1|0|0|0|	10

Here *priority* is level of interest of user to this category; 1 is the highest priority. Evidently, that 2|0|0|0|0| and 2|1|0|0|0| have in this case the lower probability to be selected,

that as a result, according to type and parameters of used distribution, provides a very small amount of usage of these queries in a selection, and in future, they will be excluded from the experiments.

5.2.1.3. Exponentially distributed frequencies of interests

Let`s assume that distribution of interests of user according to general selection − *exponential* [54], with parameter λ=1.5 in a range 0÷2. We remind that probability density of exponential distribution with parameter λ for random value x is expressed by the formula (5.1):

$$\mathbb{p}^x = \lambda e^{-\lambda x} \qquad (5.1)$$

This distribution is selected because probabilities of selection of every transaction by the user depend on his interests (priority of transaction), and there are most and least frequently used transactions. Probabilities of selection of every transaction in this case can be described as exponential distribution with certain parameters.

After sorting by level of user interests, distribution of frequencies of interests of data created for a selection is shown on Fig. 5.11.

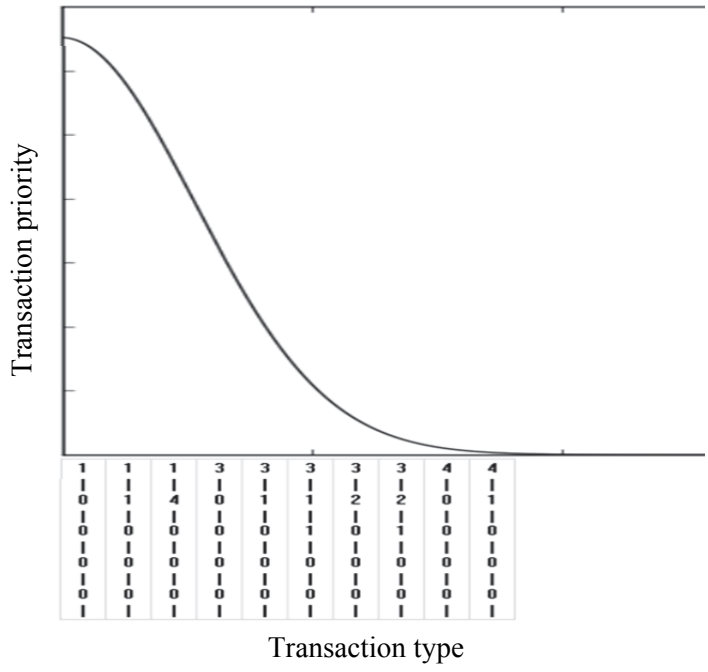


Fig. 5.11 Exponentially distributed frequencies of interests

There are researches [98] that recommend to apply lognormal distribution for description of user interests [52]. However, as priorities are shown from highest priority to

lowest priority in the considered task, bilateral lognormal distribution is less suitable than exponential.

As a result, a selection is generated, where probability to select the next element complies with standardized density value of used exponential distribution in relation to the index of element in the array of all possible identifiers of interests.

Varying the value of parameter λ, different levels of user interest to different categories of the target system can be emulated (Fig. 5.12).

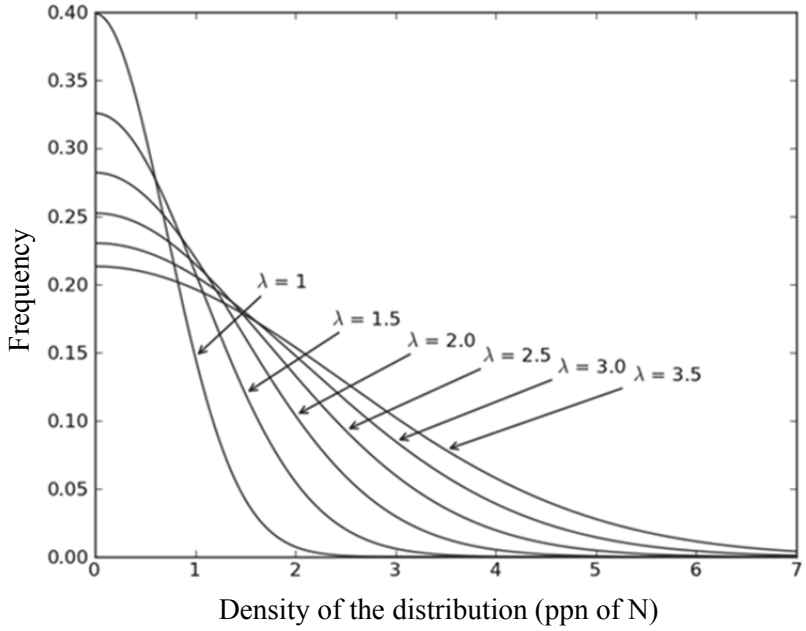


Fig. 5.12 Influence of distribution parameter to the priorities of interest

For generation of exponentially distributed interests, the following method is used (all programmatic code is shown in PHP programming language [14]).

Let`s assume:

- $s = 1.5; − value λ for the used distribution;
- $possibleKeys = array of type:
- Array (‘1|0|0|0|0|’, ‘1|1|0|0|0|’, ‘1|4|0|0|0|’, ‘3|0|0|0|0|’, ’3|1|0|0|0|’, ’3|1|1|0|0|’, ’3|2|0|0|0|’, ’3|2|1|0|0|’, ‘4|0|0|0|0|’, ‘4|1|0|0|0|’, ‘2|0|0|0|0|’, ‘2|1|0|0|0|’), that contains transaction identifiers in the order of decreasing of user interests;
- getrandmax() – the system function that returns the largest value of random number;
- M_EULER – constant, Euler`s number/ base of the natural logarithm (2,7182818…).

For calculation of index at first $x − random number in the range 0÷2 is calculated:

```
$randmax = getrandmax();
$x = 2 * rand(0, $randmax) / $randmax.
```

Then, using the formula (5.1) the value of exponential function of distribution at this point is calculated:

```
$ind = 1-pow(M_EULER, $s * $x).
```

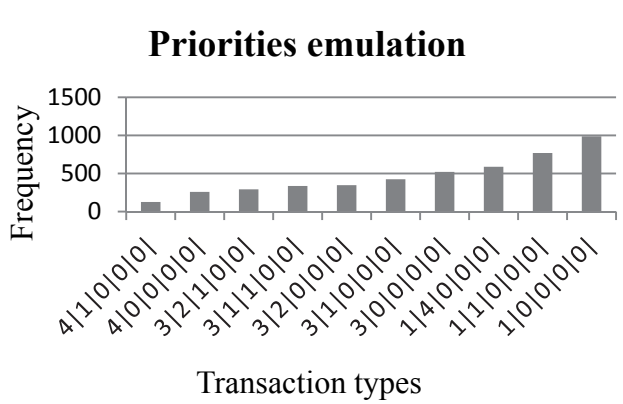
As a result, on the basis of obtained value the used index is calculated:

```
$final_ind = count( $possibleKeys ) - 2 - round( $ind * count( $possibleKeys ), 0,
PHP_ROUND_HALF_UP )-1;
$retArr[ $index ][ ] = $possibleKeys[ $final_ind ].
```

Final frequency table for selection generated in such a way is shown in Table 5.14.

Table 5.14

Frequency characteristics of selection at emulation of user interests

Transaction and its priority	*Frequency graph*	*Descriptive statistics*	
4|1|0|0|0| 125 4|0|0|0|0| 258 3|2|1|0|0| 293 3|1|1|0|0| 336 3|2|0|0|0| 347 3|1|0|0|0| 424 3|0|0|0|0| 522 1|4|0|0|0| 588 1|1|0|0|0| 769 1|0|0|0|0| 988		Mean	465
		Standard Error	81,96
		Median	385,5
		Mode	#N/A
		Standard Deviation	259,1
		Sample Variance	67178
		Kurtosis	0,51
		Skewness	0,92
		Range	863
		Count	10

Using formula (2) in language PHP because of insufficient accuracy and accuracies due to rounding, final values can differ from expected. For compliance that is more precise, it's recommend to use the following function:

$$g2d(x,\lambda) = \frac{1}{\sqrt{2*\pi*\lambda}} * e^{-\frac{1}{2}*(\frac{x}{\lambda})^2} . \qquad (5.2)$$

Formula (5.2) is presented in form of a programmatic code using PHP language:

```
function gauss2d($x,$sigma){
    return (1/sqrt(2*pi()*$sigma ))*exp(-1/2*pow(($x/$sigma),2 ) );
}
```

5.2.2. Generation of target categories complete coverage

In case of artificial generating of behavior data, there is a task to create a vector of queries, which will completely cover all possible variants of transitions within the framework of the target types of transactions. The number and structure of queries sequences created in this case have to satisfy the specified limitations:

- to cover completely all possible variants of crossing within the framework of the target types of transactions;
- each crossing has to exist only one time;
- to generate a sequence taking into consideration the current size of window (parameter *w*).
- *ct* – the number of the target transactions.

The process of generating such vector of identifiers is divided into two stages:

1. For the current value *w,* the complete set of all states − nodes of behavior graph − is created; their number – ct^2. As each node consists of *w* types of transactions, the number of elements at this stage is $w*ct^2$.

   Transition array created at this stage for the case of the target types of transactions: '1', '2' and '3' at $w = 2$ is shown on Fig. 5.13.
2. For each node of the obtained graph the sequence of crossings with all nodes is generated, including accordingly even crossing with itself.

   As a result, for the case of the target types of transactions: '1 ', '2' and '3' at $w = 2$, after completion of the second stage, the vector consisting of 162 identifiers of transactions will be created.

Final sequence will contain the vector of queries, using which for training, transitions between all interesting types of transactions will be created. Total number of the created vectors will be equal to $w*ct^4$.

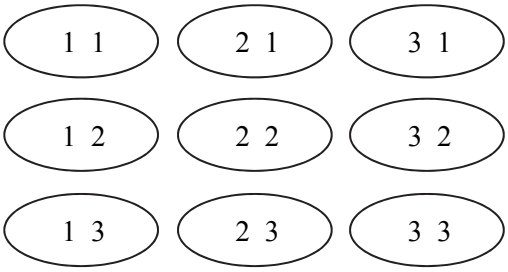


Fig. 5.13 An example of an array of transitions created at the first step of generation

Software code for generating such sequence of queries for training is shown below, if the size of window is $w = 2$ and the target types of transactions: '1', '2' and '3'.

```
var tr_names_arr=['1','2','3'];
var generated_arr = [];
var generated_str = '';
//First loop, prepare initial all nodes array
for(var i=0;i<tr_names_arr.length;i++){
        generated_arr[i] = [];
        for(var j=0;j<tr_names_arr.length;j++){
//Here now hardcoded w param used on concatenation
                generated_arr[i][j] = tr_names_arr[i]+','+tr_names_arr[j];
        }}
//Second loop, make array of all nodes interconnections
var generated_arr_final = [];
for(var i=0;i<generated_arr.length;i++){
        generated_arr_final[i] = [];
        for(var j=0;j<generated_arr[i].length;j++){
                generated_arr_final[i][j] = [];
                var current_node = generated_arr[i][j];
                for(var i2=0;i2<generated_arr.length;i2++){
                        generated_arr_final[i][j][i2] = [];
                        for(var j2=0;j2<generated_arr[i2].length;j2++){
                                generated_arr_final[i][j][i2][j2] =
[current_node,generated_arr[i2][j2]];
                                generated_str += current_node+','+generated_arr[i2][j2]+',';
                        }
                }}}
```

5.2.3. Comparison of descriptions of profiles trained on data having different parameters

The first stage of the experiment is completed − two sets of selections, emulating general behavior of class of users and interests of individual user, are created. The second stage is the following: on the basis of created data it is necessary to train two profiles and to provide a typical, as well as untypical selection for each profile for analysis; in other words, to

provide data about behavior of class for the profile of class for the analysis, as well as data about certain user behavior also created on the basis of priorities of interests.

To compare the efficiency of two profiles, each of them has been trained on the basis of two different selections created using identical values of internal parameters of algorithm. Each has been used for detection anomalous activity for all possible variants of pairs *Profile*/*data* (Table 5.15). For example, *same_exp* determines usage of profile trained on the basis of equally distributed interests for the analysis of set of transactions with exponentially distributed interests.

For construction of both profiles, identical values of internal parameters are used: $W = 3$, $Z = 1.5$ (parameters adjust the algorithm according to the features of current subject domain, W − size of window, Z − level of fine, if there is no transition between nodes describing current and previous states).

Table 5.15

Denotation of types of carried out experiments

Profile / Behavior	***Class of users***	***Personal behavior***
Class of users	same_same	same_exp
Personal behavior	exp_same	exp_exp

Every experiment consists of selection of the specified amount of transaction identifiers from selections that present target **behavior**, and calculation the value of profile metrics for every transaction. The experiment consists of three stages:

1. ***Warming up*** – is initial stage, a profile does not have enough information for adequate estimation of behavior. At this stage, values of metrics can differ significantly for every next step.
2. ***Typical behavior*** – behavior of profile is estimated in case of transactions typical for the emulated user.
3. ***Anomalous behavior*** − behavior of profile is estimated in case of transactions not typical for the emulated user.

For all experiments the amount of selections for every stage has been selected 50/200/50, for warming−up/ typical behavior/ anomalous behavior, accordingly. All graphs are shown in the ranges of ordinate 1÷1.6 for clearness.

5.2.4. Results of experiments

The overall results of all experiments at the first stage are presented in Table 5.16.

Table 5.16

Summary values of statistical characteristics of results

Property	*same_same*	*same_exp*	*exp_same*	*exp_exp*
Mean	1,36	1,42	1,37	1,14
Standard Error	0,0016	0,0014	0,0023	0,0040
Median	1,361	1,4151	1,3785	1,1158
Mode	1,5	1,5	1,5	1,5
Standard Deviation	0,0269	0,0245	0,0399	0,0680
Sample Variance	0,0007	0,0006	0,0015	0,0046
Kurtosis	7,8709	3,2380	27,0934	7,2465
Skewness	1,8651	1,8347	-4,1711	2,2546
Range	0,2037	0,1078	0,4444	0,4215
Sum	366,97	425,19	410,91	326,95
Count	269	299	299	286

- Experiment *same_same*. The profile is constructed on the basis of transactions selected from general set uniformly. Transactions for estimation by the profile are selected uniformly too.
- Experiment *same_exp*. The profile is constructed on the basis of transactions selected from general set uniformly. Transactions for estimation by the profile are selected according to an exponential law.

- Experiment *exp_same*. In this case the profile is constructed on the basis of user interests sorted out according to an exponential law of distribution. Transactions for estimation by the profile are selected with equal probability.
- Experiment *exp_exp*. In this case a base profile of behavior is constructed on the basis of user interests sorted out according to an exponential law of distribution, and transactions for estimation by the profile are selected according to an exponential law.

Summarized graph of all experiments results is shown on Fig. 5.14. Evidently, that for cases *same_same*, *same_exp* and *exp_same* behavior of profile differs not significantly, and only for the experiment *exp_exp* the difference is considerable.

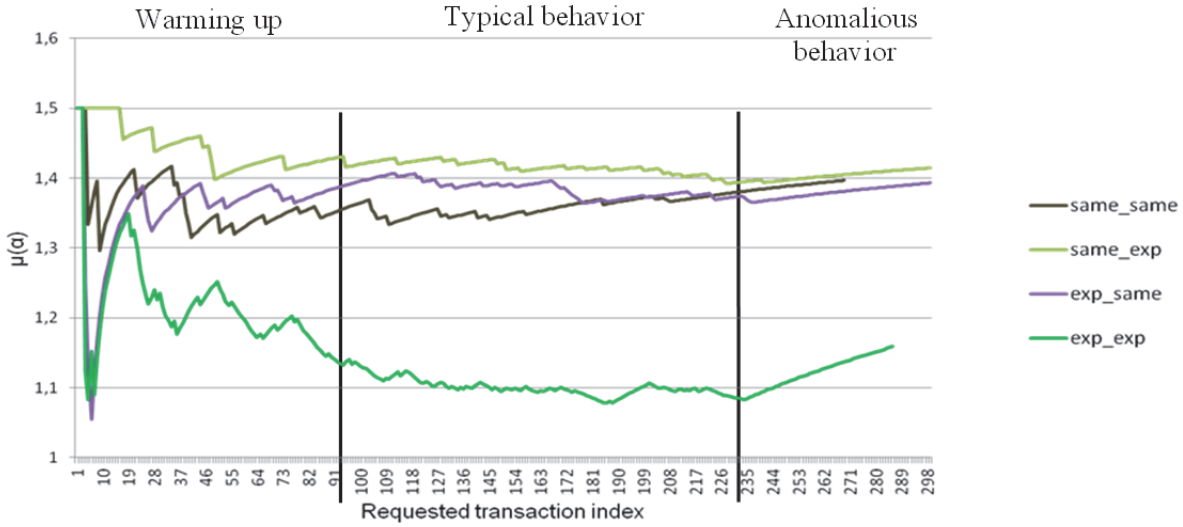


Fig. 5.14 Summarized graph of all experiments results

Such result confirms a hypothesis that personal profile will be the best classifier for such user on the basis of that behavior he has been trained. The profile *exp_exp* also shows the best increase of metrics value in the phase of anomalous behavior (last 50 values). It is a good sign showing that this configuration can detect intrusion more effective.

The level of metrics value, as well as dynamics of its change differs at the stage of "*normal*" behavior. The stage "*warming–up*" lasts longer than for other variants of test terms. We consider such behavior is correct, because the scopes of limitations for PUBP are narrower than for general, so the profile requires more time to achieve the stationary mode.

5.3. Third set of experiments

In the previous chapter, it has compared the profiles trained according to selections that differ significantly. It is also interesting to compare sensitivity of the profile trained according to selections with the same law of distribution, but different values of internal parameters.

To determine these characteristics of the profile, the following part of experiments has been conducted.

Basic purpose of the second set of experiments is to obtain the profile characteristics in different terms. It is important to understand the features of dependences of the profile work results from the values of internal variables and characteristics of data selections used for training and analysis.

- Longer interval of "*normal*" behavior is used in the terms of initial set of experiments that will allow estimating quality of the stationary mode at this stage.
- A few profiles are created on the basis of the same laws of distribution of selection frequencies with parameter $\lambda = 1.5$ and without differences in priorities of transactions.
- Models are trained according to exponentially distributed interests of user with parameter $\lambda = 1.5$, but with different priorities of transactions.
- Profiles are trained according to exponentially distributed interests of user with other value of parameter λ, but with initial priorities of transactions.

5.3.1. Stationarity of metrics value (if there is no anomaly in user behavior)

It was decided to use longer interval for "*normal*" behavior in terms of initial set of experiments that will allow estimating the stationary mode at this stage. The purpose of this set of experiments is to estimate parameters of metrics values at longer stage of "*normal*" behavior in comparison with a base set of experiments. It is important to make sure that in time, internal weight balance the values of metrics well, providing the stationary mode.

The graph of results of the experiments conducted in the above-mentioned terms is shown on Fig. 5.15. The graph line of average results is highlighted by bold green line.

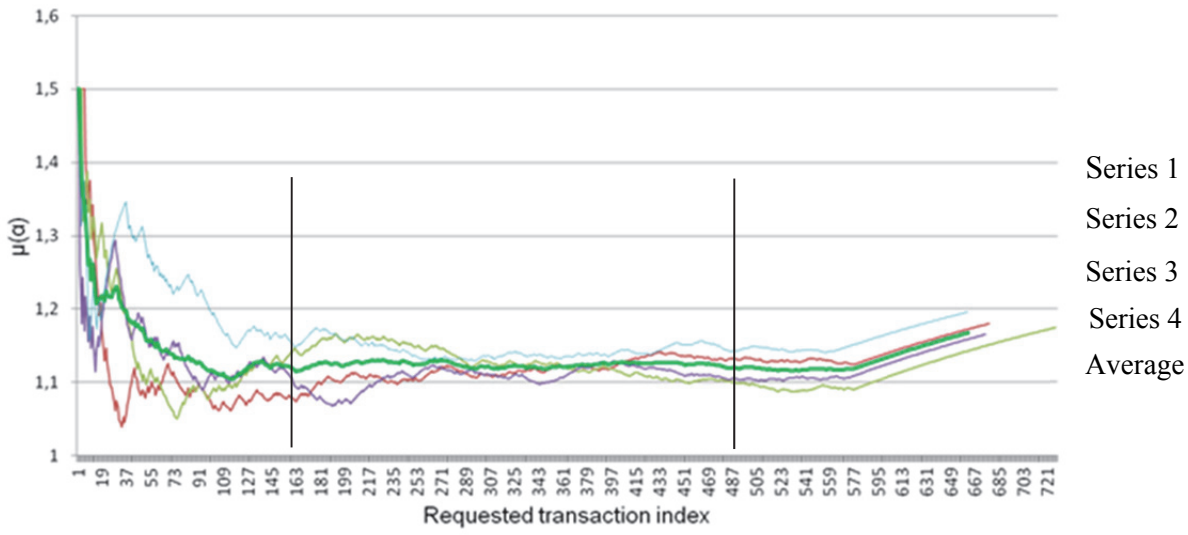


Fig. 5.15 Results of analysis by profile exp_exp a large amount of „*normal*" activities

Based on the results of the experiment 1, it is possible to assert that at the stage of "*normal*" behavior, values of metrics will aim to the stationary mode. However the level at which it will be achieved is not a constant value, so it is not possible to determine limits of metrics value, exceeding of which will detect anomalous behavior.

5.3.2. Levels of difference of metrics using same structure data about behavior

The experiment consists of creation of a few profiles based on an exponential law of distribution of selection frequencies with parameter $\lambda = 1.5$ and the same priorities of transactions.

The purpose of this experiment is the analysis of behavior of the profiles, trained according to data with identical characteristics. Properties of the profiles obtained in such way have to be similar too. It shows minimum dependence on certain values used for training of sets of transactions, because exactly a specific of user **behavior is important** for a profile, but the **sequence** of queries of transactions is **less important**.

On charts at Fig. 5.16 dynamic of change of metrics, value for two profiles trained according to the same data and with the same values of settings are shown. The graph shows that there are no fundamental differences; the results of estimation using the first and second model differ slightly.

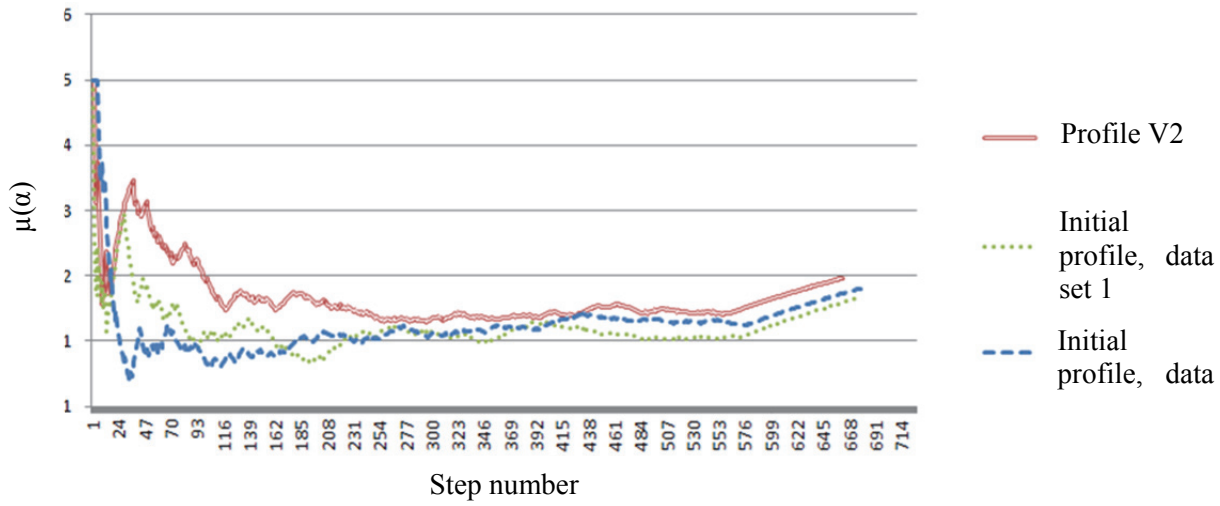


Fig. 5.16 Two profiles constructed based on the same data

The results of the experiment confirm a hypothesis that final profile does not depend on the **order** of selections used for training in case if they represent behavior of the same user.

5.3.3. Difference in metrics for different values of differences in base selections

Training of the profile according to exponentially distributed interests of user with parameter $\lambda = 1.5$, but with different priorities of transactions, is implemented. The profiles based on more and more different priorities of user interests are created. The analysis is executed according to selection with the same characteristics.

The purpose of this set of experiments is detection of the profile sensitivity to changes in person`s interests. At first minimum difference in priorities of interests is used; during the experiments the difference is increasing. The orders of changed priorities are shown on Table 5.17.

At this stage, it is important not to depend on random overshoot in the value of metrics. It is found that in previous experiments, it was interesting to determine general dynamics of behavior, but in this case, it is important to take into account even minimum differences in behavior of metrics. For this purpose, five successive experiments have been conducted at every stage, and the result of the stage is average value of metrics.

Distribution of results of the experiment for change 3 is shown on Fig. 5.17. It is presented that two groups of results split; initial, first and third in one group, and all other in other. In addition, we can see that the second group is widely spaced in comparison with the first.

Table 5.17

Variants of priorities of interests

Initial profile		*Change 1*		*Change 2*		*Change 3*	
Code of interest	Priority	Code of interest	Priority	Code of interest	Priority	Code of interest	Priority
1|0|0|0|0|	1	1|0|0|0|0|	2	1|0|0|0|0|	10	1|0|0|0|0|	1
1|1|0|0|0|	2	1|1|0|0|0|	1	1|1|0|0|0|	9	1|1|0|0|0|	2
1|2|0|0|0|		1|2|0|0|0|		1|2|0|0|0|		1|2|0|0|0|	
1|3|0|0|0|		1|3|0|0|0|		1|3|0|0|0|		1|3|0|0|0|	
1|4|0|0|0|	3	1|4|0|0|0|	3	1|4|0|0|0|	3	1|4|0|0|0|	3
2|0|0|0|0|	11	2|0|0|0|0|	11	2|0|0|0|0|	11	2|0|0|0|0|	11
2|1|0|0|0|	12	2|1|0|0|0|	12	2|1|0|0|0|	12	2|1|0|0|0|	12
3|0|0|0|0|	4	3|0|0|0|0|	4	3|0|0|0|0|	4	3|0|0|0|0|	8
3|1|0|0|0|	5	3|1|0|0|0|	5	3|1|0|0|0|	5	3|1|00|0|	7
3|1|1|0|0|	6	3|1|1|0|0|	6	3|1|1|0|0|	6	3|1|1|0|0|	6
3|2|0|0|0|	7	3|2|0|0|0|	7	3|2|0|0|0|	7	3|2|0|0|0|	5
3|2|1|0|0|	8	3|2|1|0|0|	8	3|2|1|0|0|	8	3|2|1|0|0|	4
4|0|0|0|0|	9	4|0|0|0|0|	9	4|0|0|0|0|	1	4|0|0|0|0|	9
4|1|0|0|0|	10	4|1|0|0|0|	10	4|1|0|0|0|	2	4|1|0|0|0|	10
Change 4		***Change 5***		***Change 6***		***Change 7***	
Code of interest	Priority	Code of interest	Priority	Code of interest	Priority	Code of interest	Priority
1|0|0|0|0|	2	1|0|0|0|0|	10	1|0|0|0|0|	12	1|0|0|0|0|	10
1|1|0|0|0|	3	1|1|0|0|0|	2	1|1|0|0|0|	11	1|1|0|0|0|	9
1|2|0|0|0|		1|2|0|0|0|		1|2|0|0|0|		1|2|0|0|0|	3
1|3|0|0|0|		1|3|0|0|0|		1|3|0|0|0|		1|3|0|0|0|	
1|4|0|0|0|	4	1|4|0|0|0|	8	1|4|0|0|0|	1	1|4|0|0|0|	8
2|0|0|0|0|	12	2|0|0|0|0|	11	2|0|0|0|0|	5	2|0|0|0|0|	7
2|1|0|0|0|	1	2|1|0|0|0|	6	2|1|0|0|0|	7	2|1|0|0|0|	6
3|0|0|0|0|	5	3|0|0|0|0|	4	3|0|0|0|0|	2	3|0|0|0|0|	5
3|1|0|0|0|	6	3|1|0|0|0|	5	3|1|0|0|0|	4	3|1|0|0|0|	4
3|1|1|0|0|	7	3|1|1|0|0|	12	3|1|1|0|0|	8	3|1|1|0|0|	12
3|2|0|0|0|	8	3|2|0|0|0|	7	3|2|0|0|0|	3	3|2|0|0|0|	11
3|2|1|0|0|	9	3|2|1|0|0|	3	3|2|1|0|0|	6	3|2|1|0|0|	13
4|0|0|0|0|	10	4|0|0|0|0|	9	4|0|0|0|0|	10	4|0|0|0|0|	2
4|1|0|0|0|	11	4|1|0|0|0|	1	4|1|0|0|0|	9	4|1|0|0|0|	1

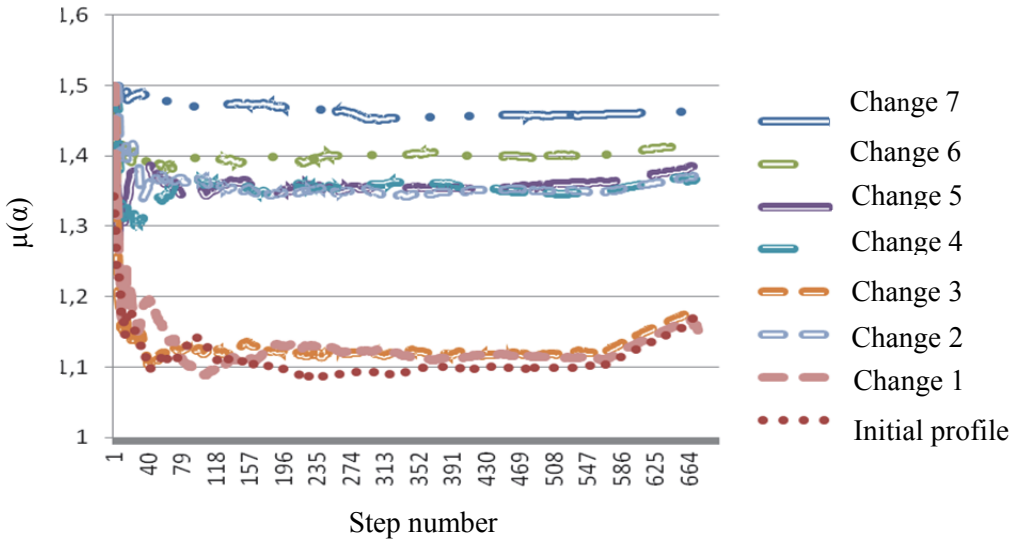


Fig. 5.17 Summarized values of metrics for the 3rd experiment

The experiment can be interpreted as successful, because dependence between level of difference from "*normal*" in the analyzed behavior and value of metrics is presented and easily visible in resulting data. The greater the difference is, the more the final value of metrics differs from "*normal*".

Grouping of results of a few experiments (change 1, change 2) in the area of "*normal*" behavior can be explained by small difference made in initial data at these stages.

5.3.4. Metrics value dependence based on distribution parameters of user interests

Training of the profile on exponentially distributed interests of user with different values of parameter λ, but with initial priorities of transactions is executed.

The purpose of experiment is to receive information about behavior of profile at the change of frequency of user interests, but without the change of priorities. It is important to know how this type of change of behavior may impact on the profile trained according to initial history of behavior.

Initial base value of parameter λ was 1.5. Four parameter λ different values have been used − 1, 1.5, 2 and 2.5. Graphs of density of exponential distribution for these parameters are shown on Fig. 5.18.

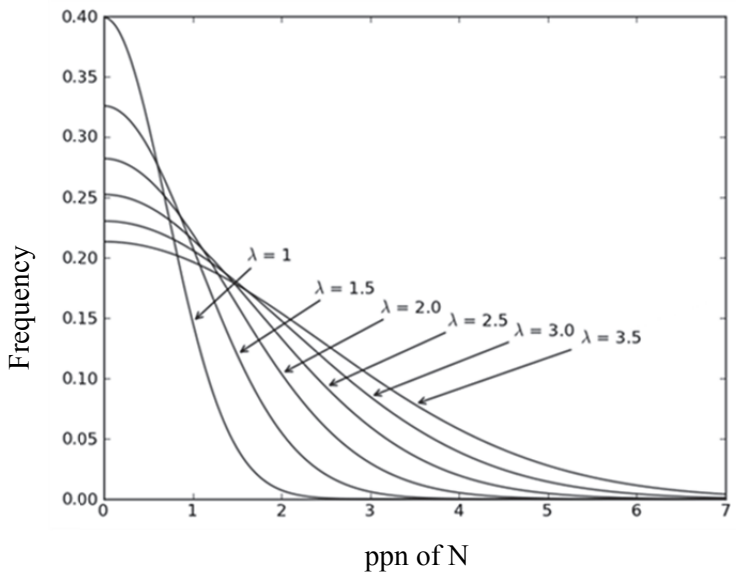


Fig. 5.18 Frequency distribution for different values λ

Accordingly, frequency distributions of indexes shall have different characteristics at different values λ. General view of distributions of interests for each parameter is shown on Fig. 5.19. In the process of implementation of experiments at this stage for each value of parameter λ five testing starts of profile have been executed. Final result is average value of metrics for all five starts.

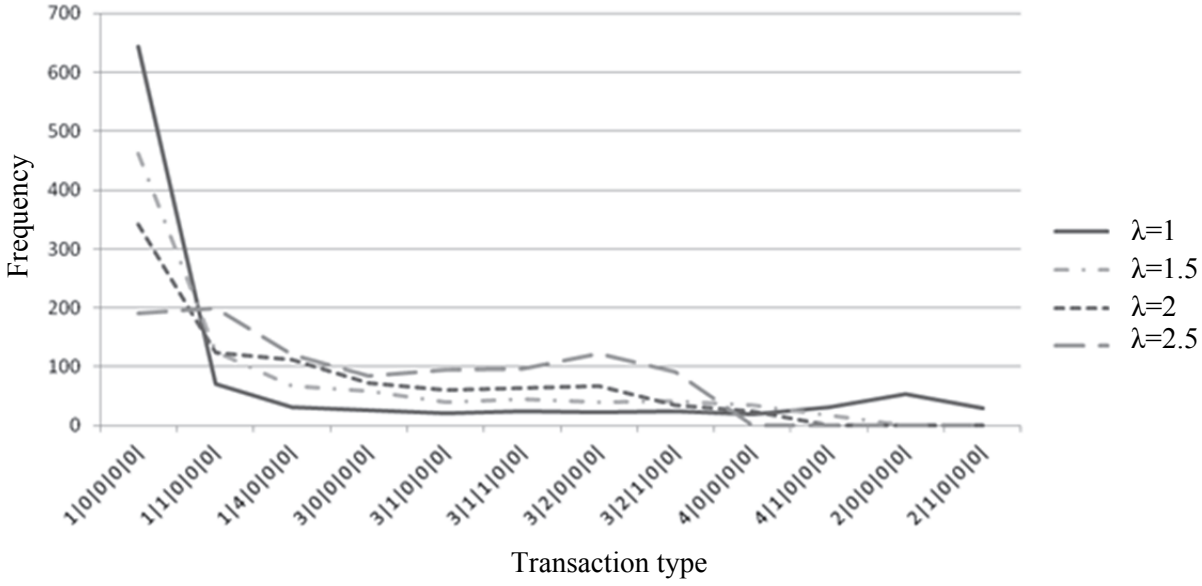


Fig. 5.19 Compliance of frequencies of interest identifiers

On Fig. 5.20 final values of metrics for different values of parameter λ are shown. Obviously, that the profile of behavior is sensitive to the change of frequency characteristics of analyzed data.

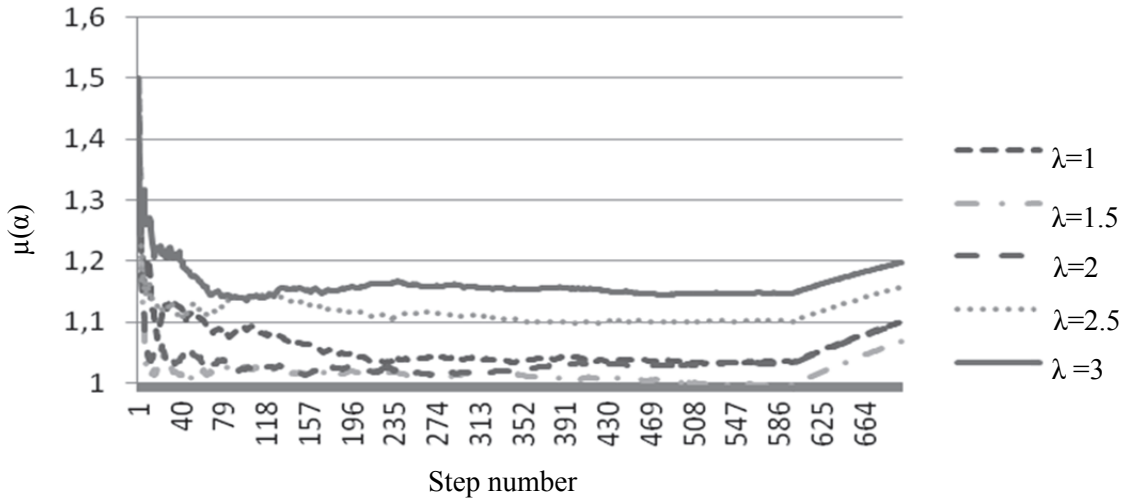


Fig. 5.20 Final average values of metrics for different values λ

5.3.5. Dependence of value of metrics on level of anomaly in training selection

In the process of research, an additional experiment has been carried out: mixing of "*normal*" and "*anomalous*" behavior at one stage with different probabilities of selection "*anomalous*" action. The results obtained differed from the expected ones, which requires further researches, and description is shown in this chapter.

The purpose of the experiment was to estimate the impact of presence of level of the anomaly on final value of metrics. For this purpose in a loop 10 iterations for 300 steps of "*normal*" behavior have been made, and on every iteration probability of selection of anomalous step increased by 0.1.

Initial training selection has been used in the experiment, as well as the profile, constructed on the basis of exponentially distributed frequencies of user interests.

On Fig. 5.21 dynamics of change of metrics is shown at increasing probability of presence anomaly. It is shown that the larger the probability of selection of anomalous step at the stage of "normal" behavior (i.e. the more often anomalous steps appear in "*normal*" selection), the slower the value of metrics increases.

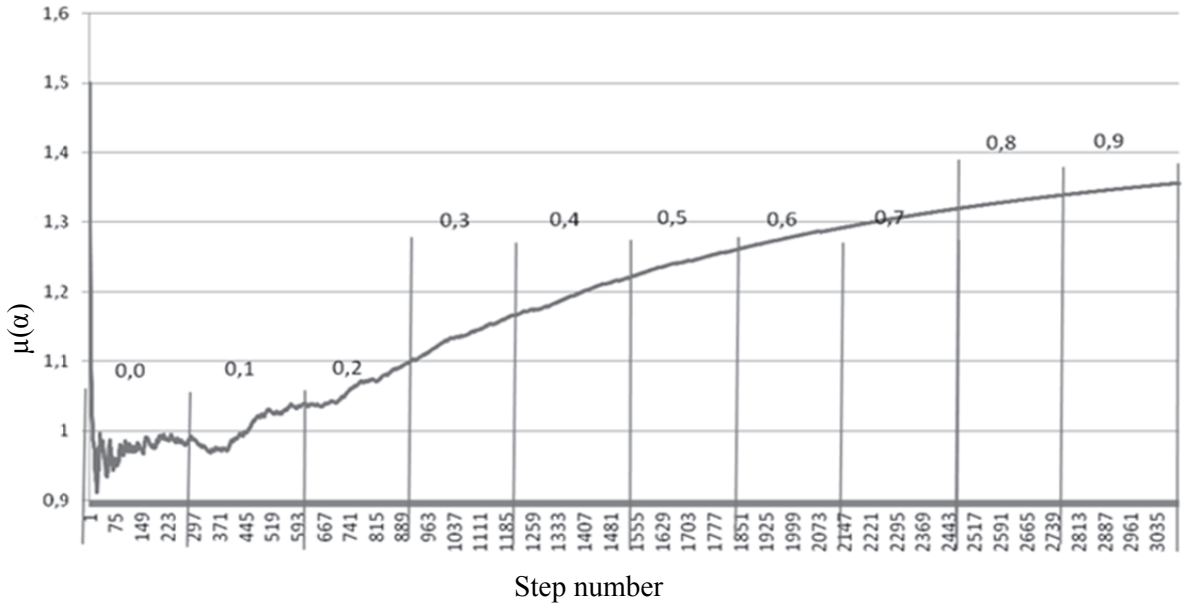


Fig. 5.21 Dynamics of increase of in metric value at different probabilities of the presence of anomalies

For a more detailed description of such dynamics of change metrics, additional set of selections has been made, in each of which the value of presence anomaly has been fixed, but increased at every step by 0.1. Results are shown on Fig. 5.22. Same as on Fig. 5.21 it is shown that by the increase of probability of anomaly presence in the analyzed behavior the value of metrics increases slower. Thus, after achievement of probability 0.4 the dynamics of change of values of metrics aims to become very similar.

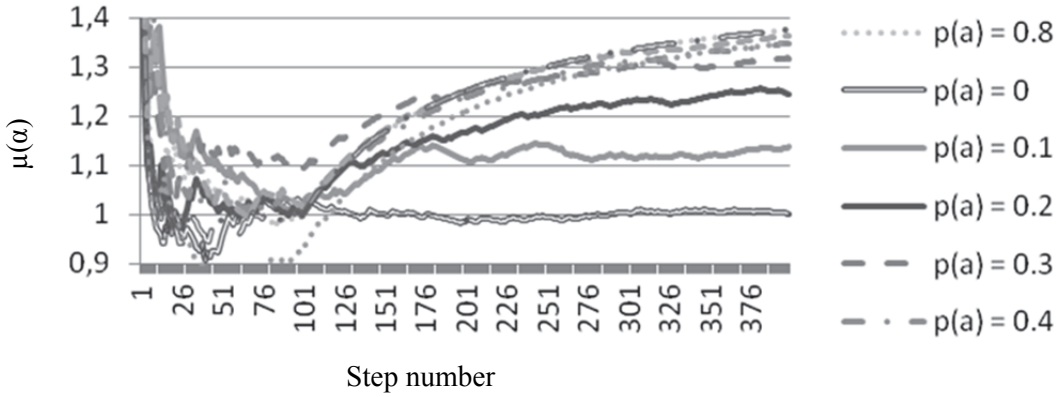


Fig. 5.22 Rate of increase in metric value at fixed values of presence of anomalies

5.4. Summary

Experimental part of this research includes a few series of experiments analyzing different descriptions of the system. All conducted experiments investigate possibilities of user anomalous behavior detection, using general user behavior profile and personal adaptive profile of user behavior.

As well as significant part of this section is devoted to description of the method developed for the generation of artificial data about user behavior having different sets of priorities in the target system.

- The use of open source programming languages allowed successful implementation of software complex on development of all required experiments.
- Using a hierarchical structure allowed effectively represent transaction types created in the experimental system.
- General approbation of possibility to use the offered approach in solving the set task confirmed such possibility.
- Comparison of the effectiveness PAUBP and CUBP system and found that in most cases PAUBP shows more accurate results in the classification of abnormal behavior.
- The methodology developed for generating artificial data on the behavior of users having different priorities allowed to successfully produce several series of experiments required.
- Comparison of the influence of the values of the internal variables of the algorithm on the classification results revealed the requirements for their values in different cases.
- Comparison of performance effects of training data on the classification results revealed important characteristics of behavior profiles.
- Comparison of the influence of the values of data to classification results also revealed important characteristics of behavior profiles.

SUMMARY AND CONCLUSIONS

Discussion and analysis of results. The main result of this research is the development of the methodology of effective determination of the anomaly level in electronic information system user behavior. The efficiency of the created approach complies with initial requirements. Introduction of the Personal Adaptive Profile of User Behavior allowed increasing accuracy of basic approach to anomalous behavior detection.

To show the efficiency, a series of different experiments representing different descriptions of the created approach have been conducted.

Analyzing the research results, it is possible to assert that the basic hypothesis set at the beginning of the thesis, which is related to the necessity to analyze user personal interests within the framework of the target system, is correct; and Personal Adaptive Profile of User Behavior is the best classifier of user behavior type in comparison with General User Behavior Profile, and more effectively determines the anomaly level of user behavior.

Within the framework of confirmation of basic hypothesis, the lower−level hypotheses set initially have also been confirmed:

Yes − the graph of Markov chain can be used for presentation of information about the features of user behavior;

Yes − there is possibility to compare efficiency of user behavior profiles;

Yes − there is possibility, using the program, to generate data about user behavior, taking into consideration his different interests within the framework of the target system.

The basic results obtained within the framework of research and development of the doctoral thesis can be set as follows.

1) An analysis of the existing modern approaches to anomalous behavior detection of users of information systems has been made. During research it has been concluded that none of the considered approaches provide complete implementation of all set requirements.
2) The existing methods of construction and formalization of user behavior based on the use of different variants of the Bayes networks, ontology engineering, mobile agents have been investigated and analyzed.
3) Possibilities of application of Markov chains in the tasks of formalization of user behavior profile have been investigated, which confirmed the correctness of the set lower−level hypothesis.
4) Basic approach for creation and use of general user behavior profile has been developed.

5) To increase efficiency of the basic approach for intrusion and anomalous behavior detection, a method using the Personal Adaptive Profile of User Behavior has been developed.
6) The methodology for assessment the efficiency of behavior profiles has been developed; it allows implementing a comparative analysis of the Personal Adaptive Profile of User Behavior and General User Behavior Profile.
7) The developed method of dynamic threshold of the anomaly level allows improving the efficiency of user classification on "*normal"* and *"anomalous"*.
8) Using the developed methodology for user anomalous behavior detection, a comparative experimental analysis of the efficiency of the Personal Adaptive Profile of User Behavior and General User Behavior Profile has been implemented.
9) Different sets of experiments as software system have been developed and realized, which allow estimating different descriptions of behavior profiles.

The obtained conclusions of the doctoral thesis are following:

1) On the basis of Markov chains it is possible to quickly create and effectively use user behavior profiles. The specifics of user behavior uniquely determines the structure of behavior profile.
2) In accordance with the methodology for estimation the efficiency of behavior profiles it is possible to compare the efficiency of different user behavior profiles.
3) Introduction of the Personal Adaptive Profile of User Behavior caused an increase in accuracy of anomalous behavior detection.
4) The results of the conducted experiments has shown that the method of the Personal Adaptive Profile of User Behavior is more effective than General User Behavior Profile.

The practical value of research and opportunities for its implementation. The results of the proposed research can be used as a basis for the creation of a large number of systems of protection against abnormal activity. Every modern complex information system that operates critical data requires reliable protection against this type of attack.

It should be clear, that all researches underlying this thesis are based on artificial data of different complexity. When using data of really working applications, additional actions may be required to increase the percentage of anomalous activities detection.

The presence of formalized representation of a typical behavior of the user (or user group) can be used not only to detect intruders, but also for the transmission of accumulated knowledge, for example, to create domestic intellectual advisers.

One more interesting possible approach is to build model of texts author style [73] to detect new texts authorship based on specific of person writing and texts building.

It is also possible to use the approach considered not within the multiuser information systems, but also for protection of personal electronic devices, when the very nature of use of the device is employed as a kind of analogue of constant electronic signature of the owner.

INFORMATION SOURCES

1. Alavi M., Leidner D. E. Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues // *MIS Quarterly.* – March 2001. – Vol. 25, No. 1. – P. 107–136. – Available on-line on: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.98.8885&rep=rep1&type=pdf. – Last accessed 2014.04.08.
2. Antoniou G., Harmelen F. Web Ontology Language: OWL *// Handbook on Ontologies / S. Staab, R. Studer (Eds.)*. – Berlin: Springer-Verlag, 2004, p. 67–92 (*Series: International Handbooks on Information Systems*)
3. *ApacheBench.* – Copyright Adam Twiss, Zeus Technology Ltd., 1996.
4. Arndt C. *Information Measures: Information and its Description in Science and Engineering.* – Springer-Verlag, 2013. – ISBN 978-3-540-40855-0. (*Springer Series: Signals and Communication Technology*)
5. Bahder T.B. *Mathematica for Scientists and Engineers*. – Addison-Wesley Publ. Company, 1995. – ISBN-10: 0201540908.
6. Baum L. E., Petrie T. Statistical inference for probabilistic functions of finite state Markov chains *// Annls of Mathematical Statistics.* – 1966. –No. 37. – P. 1554-1563.
7. Beizer B. *Black-Box Testing.* –John Wiley, 1995. – ISBN: 0471120944, 9780471120940.
8. Bernard V. L. *A Guide to Microsoft Excel 2002 for Scientists and Engineers*. – St. Francis Xavier University Nova Scotia, Canada; 2003. – 320 p. – ISBN 0 7506 5613 1.
9. Bhuyan M. H., Bhattacharyya D. K., Kalita J. K. Network Anomaly Detection: Methods, Systems and Tools // *IEEE Communications Surveys & Tutorials.* – 2013. – Vol. 16, Issue 1. – P. 303 – 336. – ISSN:1553-877X.
10. Billo E. J. *Excel for Scientists and Engineers: Numerical Methods.* 1st ed. – Wiley-Interscience, 2007. – 480 p. – ISBN-13: 978-0471387343.
11. Bishop C. M.. Pattern Recognition and Machine Learning (*Information Science and Statistics*). NY: Springer-Verlag, 2006.
12. Bongard M. *Pattern Recognition*. – SAMS, 2000. – ISBN: 0810491656.
13. Brandes U., Eiglsperger M., Lerner J., Pich C. *Graph Markup Language (GraphML)*. – CRC Press, LLC, 2004.
14. Brooks D. R. *An Introduction to PHP for Scientists and Engineers.* – Springer Science Media, 2008. – ISBN-10: 184800236X.

15. Chandola V., Banerjee A., Kumar V. Anomaly Detection: A Survey // *ACM Computing Surveys*. –2009. – No.9. – P. 1–72.

16. Chang M., Mathiske B., Smith E., Chaudhuri A., Bebenita M., Gal A., Wimmer Ch., Franz M. *The Impact of Optional Type Information on JIT Compilation Of Dynamically Typed Languages //* 7th Dynamic Languages Symposium (DLS 2011), Portland, Oregon, ACM Press, ISBN 978-1-4503-0939-4, pp. 13–24; October2011. doi:10.1145/2047849.2047853.

17. Chunfu J., Feng Y. An Intrusion Detection Method Based on Hierarchical Hidden Markov Models // *Wuhan University Jornual of Natural Sciences*. – 2007. – Vol. 12, No. 1. – P. 135-128.

18. *Cost of Cyber Crime Study: United States Benchmark Study of U.S. Companies. – Ponemon Institute,* October 2013. – Available on-line on: http://media.scmagazine.com/documents/54/2013_us_ccc_report_final_6-1_13455.pdf. – Last accessed 2014.02.15.

19. Cover T. M., Thomas J. A. *Elements of Information Theory*. – John Wiley & Sons, 1991. – Print ISBN 0-471-06259-6. Online ISBN 0-471-20061-1.

20. *Cyclone.io* project. web-page / Internet. – www.cyclone.io – Last accessed 2014.02.06.

21. *DARPA Agent Markup Language+Ontology Interface Layer / Internet*. – http://www.daml.org/2001/03/daml+oil-index. – Last accessed 2014.10.11.

22. Dasgupta D., Gonsalez F., Yallapu K., Gomez J., Yarramstetti R.. CIDS: An agent-based intrusion detection system // *Computers & Security.* – 2005. – Vol. 24. – P. 387-398.

23. Day J. D., Zimmermann H. The OSI Reference Model // *Proceedings of the IEFJ2.* – 1983. – Vol. 71, No. 12.

24. Downey A. B. *How to think like a computer scientist. C++ Version*. 2012 / Internet. – http://www.xplora.org/downloads/Knoppix/books/Open_Book_Project/thinkCScpp.pdf. – Last viewed 2014.09.30.

25. DTI 2002. *Information Security Breaches Survey 2002*. Technical report. Department of Trade & Industry, April 2002. URN 02/318. – Available on-line on: http://www.vicomsoft.com/downloads/learning/dti_security_survey.pdf. – Last accessed 2013.10.12.

26. Duval T., Jouga B., Roger L. The Mitnick Case: How Bayes Could Have Helped // *IFIP International Federation for Information Processing.* – 2005. – Vol. 194/2005. – P. 91-104. – DOI: 10.1007/0-387-31163-7_8.

27. EGEE – *Enabling Grids for E Enabling Grids for E-sciencE* / Internet. – http://www.eu-egee.org/. – Last accessed 2012.04.24.

28. Elliotte R.H., Means W.S. *XML in a Nutshell*. 3rd Edition. – O'Reilly Media; 2009. – Print ISBN: 978-0-596-00764-5.

29. Fettig A. *Twisted Network Programming Essentials.* – O'Reilly Media, 2005. – 238 p. – Print ISBN: 978-0-596-10032-2. ISBN 10: 0-596-10032-9.

30. Fine S, Singer Y, Tishby N. The Hierarchical Hidden Markov Model: Analysis and Applications // *J. Machine Learning*. – 1998. – Vol. 32, No. 1. – P.41-62.

31. Foster I. *The Grid: Blueprint for a New Computing Infrastructure*. – Morgan Kaufmann Publishers, 1999. – ISBN 1-55860-475-8.

32. *GILDA* virtual laboratory / Internet. – https://gilda.ct.infn.it/. – Last accessed 2012.04.24.

33. Gray J. The Transaction Concept: Virtues and Limitations // *Proceedings of the 7th International Conference on Very Large Databases,* September 9-11, 1981, Cannes, France. – IEEE Press, 1981. – P. 144–154,

34. Gray R. M. *Entropy and Information Theory.* 2nd ed. – New York: Springer US, 2011. – 409 p. – ISBN 978-1-4419-7970-4 (Online).

35. Grinstead C. M., Snell J. L. *Introduction to Probability.* 2nd ed. – American Mathematical Society, 1997. – 510 p. – ISBN-10: 0-8218-9414-5, ISBN-13: 978-0-8218-9414-9.

36. Gunter T. D; Nicolas P. T. The Emergence of National Electronic Health Record Architectures in the United States and Australia: Models, Costs, and Questions // *Journal of Medical Internet Research.* – 2005. Vol. 7, No. 1. – Doi: 10.2196/jmir.7.1.e3. PMC 1550638. PMID 15829475.

37. Hahn B.H., Valentine D.T. *Essential MATLAB for Engineers and Scientists*. 4th edition. – Elsevier Academic Press, 2009. – 416 p. – ISBN: 978-0-12-374883-6.

38. Hahn B.H. *Fortran 90 for Scientists and Engineers*. – London: Butterworth-Heinemann, Cambridge University Press, 1994. ISBN-10: 0-340-60034-9.

39. Harary F. *Graph theory.* – Addison-Wesley Publishing, 1969.

40. Hassanzadeh O. *Introduction to Semantic Web Technologies & Linked Data*, University of Toronto, CS 443: Database Management Systems – Winter 2011. – Available on-line on: http://www.cs.toronto.edu/~oktie/slides/web-of-data-intro.pdf. – Last accessed 2014.04.08.

41. Hawkins D. M. The Problem of Overfitting *// J. Chem. Inf. Comput. Sci*. – 2004. – Vol. 44. – P. 1-12.

42. Haykin S. *Neural Networks: A Comprehensive Foundation*. – New Jersey: Prentice Hall, 1999.

43. *EHR – Electronic Health Records: Manual for Developing Countries*. – WHO Library Cataloguing in Publication Data; World Health Organization, 2006. – ISBN 92 9061 2177. – Available on-line on: http://www.wpro.who.int/publications/docs/EHRmanual.pdf. – Last accessed 2014.05.27.

44. Hill T., Lewicki P. *STATISTICS: Methods and Applications*. – StatSoft. Tulsa. 2007. – Available on-line on: http://www.statsoft.com/textbook/.

45. IMS - *Learner Information Packaging Information Model Specification, Final Specification*. Version 1.0 / Internet. – http://www.imsglobal.org/profiles/lipinfo01.html. – Last accessed 2012.04.24.

46. Isaza G., Castillo A., López M., Castillo L. Towards Ontology-Based Intelligent Model for Intrusion Detection and Prevention // *Advances in Soft Computing*. – 2009. – Vol. 63. – P. 109-116. – DOI: 10.1007/978-3-642-04091-7_14.

47. Ivanov Y, Bobick A. Recognition of Visual Activities and Interactions by Stochastic Parsing // *J. IEEE Trans on Pattern Analysis and Machine Intelligence*. – 2000. – Vol. 22, No. 8. – P. 852-872.

48. Jaiganesh V., Mangayarkarasi S., Dr. Sumathi P. Intrusion Detection Systems: A Survey and Analysis of Classification Techniques // *International Journal of Advanced Research in Computer and Communication Engineering*. – 2013. – Vol. 2, Issue 4. – P. 1629–1635. – ISSN (Print): 2319-5940, ISSN (Online): 2278-1021.

49. Jha S., Kruger L., Kurtx T., Lee Y., Smith A. *A Filtering Approach To Anomaly and Masquerade Detection*. 2005. Technical report, Univ of Wisconsin, Madison.

50. Jha S., Tan K., Maxion R.A. Markov Chains, Classifiers and Intrusion Detection // *Computer Security Foundations Workshop* (CSFW), 2001. Proceedings. 14th IEEE, vol. 1, p. 206-219, DOI: 10.1109/CSFW.2001.930147. http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7408 Last accessed 2014.02.12

51. Joaquim P. M. *Applied Statistics Using SPSS, STATISTICA, MATLAB and R.* 2nd ed. – Springer Press, 2007. ISBN 978-3-540-71972-4.

52. Johnson N.L., Kotz S., Balakrishnan N. *Continuous univariate distributions*. Vol. 1, 2nd ed. – New York: John Wiley & Sons, 1994. – ISBN 978-0-471-58495-7. (Wiley Series in Probability and Mathematical Statistics: *Applied Probability and Statistics)*.

53. Judea P. Causal inference in statistics: An overview // *Statist. Surv.* – 2009. – Vol. 3. – P. 96-146. – Doi:10.1214/09-SS057. http://projecteuclid.org/euclid.ssu/1255440554.

54. Kallenberg O. *Foundations of Modern Probability*, 2nd ed. – Springer-Verlag, 2002. – 650 p. (Springer Series in Statistics). – ISBN 0-387-95313-2.

55. J. Kopena and W. C. Regli, "*DAMLJessKB: A tool for reasoning with the semantic web*," IEEE Intelligent Systems, Vol. 18, pp. 74–77, 2003.

56. Laskey, K.B., Alghamdi, G., Wang, X., Barbara, D., Shackleford, T., Wright, E., Fitzgerald, J. Detecting Threatening Behavior Using Bayesian Networks // *Proceedings of the Conference on Behavioral Representation in Modeling and Simulation*, 2004.

57. Lawson, T. *A Conception of Ontology*. – University of Cambridge, 2004.

58. Liepins G.E. and Vaccaro H.S. Anomaly Detection: Purpose and Framework // *Proceedings of the 12th National Computer Security Conference*, October 1989. P. 495-504.

59. Lucks J.B. *Python - All a Scientist Needs*. – Pycon, 2008. – arXiv:0803.1838v1.

60. M´etivier F. *Scientific Relational Databases using MySQL and Python:* Lecture notes. – Paris: Institut de physique du globe de Paris & Universit´e Paris Diderot, 2014. – 53 p.

61. Manavoglu E., Pavlov D., Lee C. Probabilistic User Behavior Models // *Proceedings of the Third IEEE International Conference on Data Mining* (ICDM'03), November 2003, Melbourne, Florida. – IEEE, 2003, p. 203–210.

62. Markov A. A. *Theory of Algorithms.* – M.: 1954. [Translated by Jacques J. Schorr-Kon and PST staff] Imprint Moscow, Academy of Sciences of the USSR, 1954 [Jerusalem, Israel Program for Scientific Translations, 1961; available from Office of Technical Services, United States Department of Commerce] Added t.p. in Russian Translation of Works of the Mathematical Institute, Academy of Sciences of the USSR, v. 42. Original title: Teoriya algorifmov. [QA248.M2943 Dartmouth College library. U.S. Dept. of Commerce, Office of Technical Services, number OTS 60-51085.].

63. Maxfield B. *Essential MATHCAD for Engineering, Science and Math*. – London: Elsevier Academic Press, 2009. – ISBN: 978-0-12-374783-9.

64. Mea D.V. What is e-Health: The death of telemedicine*? // Journal of Medical Internet Research. – 2001. – Vol. 3, No.2.* – Doi:10.2196/jmir.3.2.e22.

65. Millman, K.J., Aivazis M. Python for Scientists and Engineers. University of California, Berkeley // *IEEE Computational Science & Engineering.* – 2011. Vol. 13, Issue 2. – P. 9–12. – ISSN: 1521-9615.

66. Mun G. J., Kim Y. M., Kim D. K., Noh B. N. Network Intrusion Detection Using Statistical Probability Distribution // Computational Science and Its Applications - ICCSA 2006. –Springer-Verlag Berlin Heidelberg, 2006, p. 340–348, (Lecture Notes in Computer Science, Vol.3981),

67. *Networkx / Internet*. – http://networkx.lanl.gov/. – Last viewed at 2014-02-07.

68. Noy N. F., McGuinness D. L. *Ontology Development 101: A Guide to Creating Your First Ontology.* KSL Technical Report, – Stanford University, Stanford, CA, 94305, 2006. – Availible on-line: http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html. – Last viewed at 2015-01-07.

69. Orwant J. *Computer Science & Perl Programming: Best of TPJ*. 1st edition. – O'Reilly Media: 2002. – ISBN-10: 0596003102.

70. Osipov P., Borisov A. Advantages of Deferred Approach for Time-Critical Tasks // *Informatica*. – 2014. – Vol. 25, No. 3. – P. 467–484. – DOI: http://dx.doi.org/10.15388/Informatica.2014.24. Cited in: **ACM, DBLP, EBSCO, SCOPUS, INSPEC, IAOR, Cambridge Scientific Abstracts, Mathematical Reviews, MathSciNet, Science Citation Index Expanded, Web of Science.**

71. Osipov P. A., Borisov A. N. System for anomalous activity detection based on Markov models // *Automatic Control and Computer Sciences.* – 2011. – Vol. 45, No. 2. – P. 46–60. Cited in: **SpringerLink, SCOPUS, Academis OneFile, DBLP, Inspec**.

72. Osipov P. A., Mrochko A. E. and Borisov A. N. Identification of Differences of User Behavior Profiles and User Class Templates // *Automatic Control and Computer Sciences.* – 2014. – Vol. 48, No. 2. – P. 65–79. Cited in: **SpringerLink, SCOPUS, Academis OneFile, DBLP, Inspec**.

73. Osipov P., Rinkevics A., Kuleshova G., Borisov A. Markov chains in the task of author's writing style profile construction // *Scientific Journal of Riga Technical University, Information Technology and Management Science*. – 2014. - Vol. 17, RTU, Riga, P. 119-125. Cited in: **EBSCO, Google Scholar, Ulrich's International Periodicals Directory, VINITI.**

74. Osipov P., Borisov A. Simulation of Typical Behavior User using Markov Models // *Proceedings of 2011 Baltic Congress on Future Internet and Communications (BCFIC 2011), 15-18 February 2011, Riga, Latvia.* – Riga: Transport and Telecommunication Institute, 2011. – P. 229–236.

75. Osipov P. A. Borisovs A. Identification of Transaction Types using standard Clinical Document Architecture // Proceedings of XVI International Youth Forum „Radio Electronics and Youth in XXI century", 17-19 April 2012, Kharkov, Ukraine. Vol. 6. – Kharkov: Kharkov National University of Radio Electronics, 2012. – P. 150–151.

76. Osipov P. A., Borisov A. N. eHealth System Anomaly Activity Detection, Based on User Behavior Model // Modeling and Analysis of Safety and Risk in Complex Systems: Proceedings of International Scientific School MA SR – 2011 (Saint-Peterburg, Russia, 28 June –02 July 2011). – SPb.: SUAI, SPb., 2011. – P. 405–412.

77. Osipovs P., Borisovs A. Approaches to the Construction of Behavioural Patterns of Information System Users // *Scientific Journal of Riga Technical University. Information Technology and Management Science.* – 2012. – Vol. 15. – P.176–182. Cited in: **EBSCO, CSA/ProQuest, VINITI**.

78. Osipovs P., Borisovs A. Approaches to the Creation of Behavioural Patterns of Information System Users // *Scientific Journal of Riga Technical University.*

Information Technology and Management Science. – 2012. – Vol. 15. – P.58–64. Cited in: **EBSCO, CSA/ProQuest, VINITI**.

79. Osipovs P., Borisovs A. Non-Signature-Based Methods for Anomaly Detection // *Scientific Journal of Riga Technical University. Series 5. Computer Science, Information Technology and Management Science.* – 2010. – Vol. 44. – P. 106–110. Cited in: **EBSCO, CSA/ProQuest, VINITI**.
80. Osipovs P., Borisovs A. Usage of Ontologies in Systems of Data Exchange // *Scientific Journal of Riga Technical University. Series 5. Computer Science, Information Technology and Management Science.* – 2009. – Vol. 40. – P. 108–116. Cited in: **EBSCO, CSA/ProQuest, VINITI**.
81. Osipovs P., Borisovs A. Use of Deferred approach in Scientific Applications // *Scientific Journal of Riga Technical University. Series 5. Computer Science, Information Technology and Management Science.* – 2011. – Vol. 49. – P.139–144. Cited in: **EBSCO, CSA/ProQuest, VINITI**.
82. Osipovs, P., Borisovs, A. Practice of Web Data Mining Methods Application // *Scientific Journal of Riga Technical University. Series 5. Computer Science, Information Technology and Management Science.* – 2009. – Vol. 40. – P. 101–107. Cited in: **EBSCO, CSA/ProQuest, VINITI**.
83. Pearl, J. (1985). Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning (UCLA Technical Report CSD-850017). *Proceedings of the 7th Conference of the Cognitive Science Society*, University of California, Irvine, CA. pp. 329–334. Retrieved 2009-05-01.
84. Phyo A. H., Furnell S. M. A Detection-Oriented Classification of Insider IT Misuse // *Proceedings of the 3rd Security Conference, Las Vegas, USA*, 14-15 April 2004.
85. Pinheiro C., Carlos A. R. *Social Network Analysis in Telecommunications*. – John Wiley & Sons, 2011. – 304 p. – ISBN 978-1-118-01094-5.
86. Pokorny J. NoSQL databases: a step to database scalability in web environment // *International Journal of Web Information Systems.* – 2013. – Vol. 9, No. 1. – P. 69-82. – DOI 10.1108/17440081311316398.
87. Prechelt L. An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl for a search/string-processing program. Fakultat fur Informatik. Universitat Karlsruhe. *Technical Report* 2000-5. March 10, 2000. – Available on-line on http://www.inf.fu-berlin.de/inst/ag-se/teaching/V-EMPIR-2014/doc/jccpprtTR.pdf. – Last accessed 2014-05-27.
88. Razmerita L. Modeling Behavior of Users in Adaptive and Semantic-enhanced Information Systems: The role of a User Ontology // *Proceedings of 5th International Conference of Adaptive Hypermedia and Adaptive Web-Based Systems and Authoring*

of Adaptive and Adaptable Hypermedia workshop, P. 69-77. - 29 of July-1 August 2008, Hanover. *Internet*. - http://www.academia.edu/3134137/Modeling_behavior_of_users_in_adaptive_and_semantic-enhanced_information_systems_The_role_of_a_user_ontology Last accessed 2014.09.08

89. Redis / Internet. – http://redis.io/topics/faq. – Last accessed 2014.04.01.

90. Reza F.M. *An Introduction to Information Theory*. – New York: Dover Publications, Inc., 1994. – ISBN 0-486-68210-2.

91. RFC7159; *The JavaScript Object Notation (JSON) Data Interchange Format / Internet.* – http://tools.ietf.org/html/rfc7159. – Last accessed 2013.05.03.

92. Samek M. *Practical UML Statecharts in C/C++: Event-Driven Programming for Embedded Systems*. – CRC Press, Newnes 2008. – 728 p. – ISBN-10: 0750687061; ISBN-13: 978-0750687065.

93. Scarfone K., Mell P. *Guide to Intrusion Detection and Prevention Systems (IDPS): Recommendations of the National Institute of Standards and Technology*. NIST Special Publication 800-94. – Computer Security Division, Information Technology Laboratory, *National Institute of Standards and Technology*, 2007.

94. Seung-Hyun K., Kyong H.K., Jong K., Sung-Je H., Sangwan K. Workflow-Based Authorization Service in the Grid // *Journal of Grid Computing.* – 2004. – No. 2. – P. 43–55.

95. Shelestov, S. Skakun, N. Kussul. Agent-based approach to implementing a model of user behavior Grid-systems // Proceedings of Space Research Institute NASU-NSAU «*Informatyka, kibernetyka ta obchyslyuval'na tekhnika*», Issue. 9 (132). – Donetsk: DonNTU, 2008. – P. 8-14. – ISSN: 1996-1588.

96. Sheskin D. *Handbook of Parametric and Nonparametric Statistical Procedures*. – CRC Press, 2004. – P. 54. – ISBN 1584884401.

97. Shingo T., Susumu D., Shinji S. A user-oriented secure file system on the Grid // *The 3rd IEEE/ACM Int. Symp. on Cluster Computing and the Grid* (CCGrid 2003), Conference Report / Oral Presentation. May, 2003.

98. Shirai K. Interest rate risk modeling using extended lognormal distribution with variable volatility // *Stochastic Modeling*. – International Actuarial Association May 2010. – ISBN: 978-0-9813968-2-8.

99. Shneiderman B. The Relationship Between COBOL and Computer Science // *American Federation of Information Processing Societies* 1985. – AFIPS 0164-l 239/85/040348-352$01 .OO/OO.

100. Sriparasa S. S. *JavaScript and JSON Essentials*. – O'Reilly Media, 2013. – ISBN 10:1-78328-604-0.

101. Tabini M. PHP as a General-Purpose Language // *Linux Journal.* – Aug 18, 2004. Internet - http://www.linuxjournal.com/article/6627 Last accessed 2014.10.10

102. *Tornadoweb*. project web-page / Internet. – www.tornadoweb.org. – Last accessed 2014.02.06.

103. Turban E., Aronson J. E., Liang T. P. *Decision Support Systems and Intelligent Systems*. 7th ed. – Prentice Hall, 2005.

104. Undercoffer J., Pinkston J., Joshi A., Finin T. A Target-Centric Ontology for Intrusion Detection // IJCAI-03 *Workshop on Ontologies and Distributed Systems*, Morgan Kaufmann Pu, P. 47-58. - Acapulco, 9 August 2003. Internet. - http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.13.727
Last accessed 2014.10.01

105. Wood R. *C Programming for Scientists and Engineers*. – Penton Press, 2002. – ISBN: 1 8571 8030 5.

106. Zinky J., Shapiro R., Siracuse S., Wright T. Experience with Dynamic Crosscutting in Cougaar // *Lecture Notes in Computer Science*. – 2010. – Vol. 4803. – P. 595–612. – DOI: 10.1007/978-3-540-76848-7_41.

107. Осипов П. А., Борисов А. Н. Система обнаружения аномальных действий на основе моделей Маркова // *Автоматика и вычислительная техника.* – 2011. – № 2. – С. 46–60. Cited in: **SpringerLink, Ulrich's International Periodicals Directory, VINITI**.

108. Осипов П. А., Мрочко А. Е., Борисов А. Н. Идентификация отличий профиля поведения пользователя и шаблона класса пользователей // *Автоматика и вычислительная техника.* − 2014. − № 2. − С.5 – 24. Cited in: **SpringerLink, Ulrich's International Periodicals Directory, VINITI**.