UDC 004.652.5

Web based application for operational loss collection and value-at-risk and expected shortfall calculation

Sergey G. Shorokhov, Victoria V. Khaptakhanova

[*] *Department of Information Technology*
Peoples' Friendship University of Russia (RUDN University)
6 Miklukho-Maklaya St, Moscow, 117198, Russian Federation

Email: `shorokhov-sg@rudn.ru`, `vkhaptakhanova@mail.ru`

We study some issues of operational risk information system design and implementation in compliance with new requirements of Basel Committee on Banking Supervision and Bank of Russia. To meet regulatory requirements, banks need to create and regularly update an analytical database on operational risk losses extending back for some years. We provide requirements to the form and contents of the information to be registered for operational risk events and losses and list mandatory classifiers for risk event database. The structure of tables in risk event database is presented on entity-relationship diagram. Operational risk information system can be rapidly developed as a distributed system using web-framework Django. Such Django features as MVC (Model-View-Controller) pattern and ORM (Object-Relational Mapping) speed up the development of web-based application for risk event database and allow to avoid manipulating sophisticated SQL expressions. We provide examples of Django classes in Python for model component of Django risk event database application. We also develop and present algorithm for Value-at-Risk and Expected Shortfall calculation using data in operational risk event database. The algorithm can be easily implemented in Python using Django database-abstraction API.

Key words and phrases: operational risk, Model-View-Controller, Object-Relational Mapping, Django framework, Value-at-Risk, Expected Shortfall.

1. Introduction

Basically, operational risk is the risk of losses due to unreliability of the internal management procedures, negligence of employees, failure of information systems or the impact of external events [1]. Along with market and credit risks, operational risk is regarded as one of the most material financial risks and credit institutions worldwide are obliged to establish a risk and capital management system to detect, evaluate, and aggregate operational risk and other most material risks and assess their capital adequacy [2, 3].

In 2017, Basel Committee on Banking Supervision (BCBS) finalized the Basel III framework [4], updating minimum capital requirements for operational risk and criteria for identification, collection and treatment of internal loss data. In 2018, Bank of Russia released draft regulation [5] on the requirements to the operational risk management.

We study some methodological and practical aspects of operational risk information system design and implementation and subsequent calculation of operational risk indicators Value-at-Risk and Expected Shortfall using available data on operational losses occurred.

2. Operational risk event database structure

According to [1] each credit institution creates and maintains the analytical database on losses incurred due to operational risk events (event database). The event database shall contain information on the amount and type of losses, the dates of event occurrence, recognition and completion, event circumstances and other relevant information [6].

According to the requirements for maintaining the event database [5] (ch. 6), the event database should contain, in particular, the following information:

– unique serial identification number of the event;
– identifier of the group of homogeneous events (if any);
– date and time of event occurrence;
– date and time of event identification;
– date and time of event completion;
– event status;
– business unit, associated with the event;
– detailed description of the event;
– loss event source (cause) codes, including the code of main source (cause);
– loss event type code;
– additional (level 2) loss event type code;
– operational risk type code;
– other banking risk type codes (if any);
– business line (or business process level 1) code;
– business process (level 2) code;
– code of the bank information system used when the event occurred;
– operational risk loss data.

The following classifiers should be used when recording operational loss events:

– classifier of loss event sources (causes) [5] (clause 2.3);
– classifier of operational risk types [5] (clause 2.3.5);
– classifier of loss event types (level 1) [5] (clause 2.7);
– classifier of loss event types (level 2) [5] (addendum 1);
– classifier of business lines (level 1) [5] (clause 2.10);
– classifier of business lines (level 2) or business processes according to internal documents of the credit institution or, for instance, [7];
– classifier of direct and indirect operational losses [5] (clause 2.11);
– classifier of business units of the credit institution [5] (clause 6.6);
– classifier of other types of banking risks [5] (clause 6.6);
– classifier of information systems of the credit institution [5] (clause 6.6).

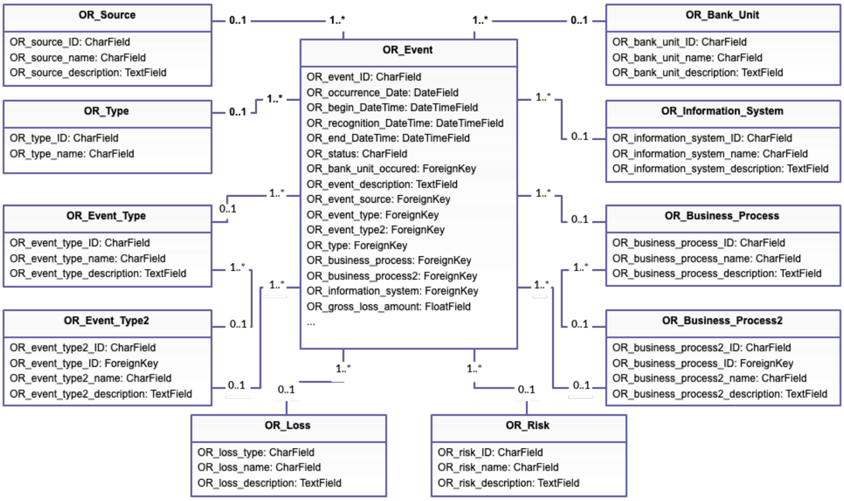
OR_Source
OR_source_ID: CharField
OR_source_name: CharField
OR_source_description: TextField
OR_Type
OR_type_ID: CharField
OR_type_name: CharField
OR_Event_Type
OR_event_type_ID: CharField
OR_event_type_name: CharField
OR_event_type_description: TextField
OR_Event_Type2
OR_event_type2_ID: CharField
OR_event_type_ID: ForeignKey
OR_event_type2_name: CharField
OR_event_type2_description: TextField
OR_Event
OR_event_ID: CharField
OR_occurrence_Date: DateField
OR_begin_DateTime: DateTimeField
OR_recognition_DateTime: DateTimeField
OR_end_DateTime: DateTimeField
OR_status: CharField
OR_bank_unit_occured: ForeignKey
OR_event_description: TextField
OR_event_source: ForeignKey
OR_event_type: ForeignKey
OR_event_type2: ForeignKey
OR_type: ForeignKey
OR_business_process: ForeignKey
OR_business_process2: ForeignKey
OR_information_system: ForeignKey
OR_gross_loss_amount: FloatField
...
OR_Bank_Unit
OR_bank_unit_ID: CharField
OR_bank_unit_name: CharField
OR_bank_unit_description: TextField
OR_Information_System
OR_information_system_ID: CharField
OR_information_system_name: CharField
OR_information_system_description: TextField
OR_Business_Process
OR_business_process_ID: CharField
OR_business_process_name: CharField
OR_business_process_description: TextField
OR_Business_Process2
OR_business_process2_ID: CharField
OR_business_process_ID: ForeignKey
OR_business_process2_name: CharField
OR_business_process2_description: TextField
OR_Loss
OR_loss_type: CharField
OR_loss_name: CharField
OR_loss_description: TextField
OR_Risk
OR_risk_ID: CharField
OR_risk_name: CharField
OR_risk_description: TextField
0..1
1..*

Figure 1. Operational risk event database structure

Thus, the event database should contain the main table with detailed information on operational risk event and loss and classifiers for the event table fields (Fig. 1).

3. Implementation of operational risk event database application

Operational risk event database should embrace all divisions of the bank, including geographically remote branches, and risk event collection application should be designed and implemented as a distributed system.

At present web development is essentially done in Python language [8] and Django is the core Python framework for creating database-driven websites [9]. Some of the well-known Django websites include Instagram, Pinterest, Dropbox, Spotify.

Django framework is an example of Model–View–Controller (MVC) architecture, first introduced in Smalltalk programming environment [10], which allows developers to change the visual part and the business logic of an application separately, without affecting one another [11]. Actually, Django architecture is called sometimes as Model–View–Template (MVT), because there are three independent layers (Model, View, and Template) in Django, responsible for managing different parts of an application (Fig. 2). Communication between the layers is possible only via an application programming interface (API).

Model component is the single source of information about application data in Django. It contains separate model (Python class) for each table (entity) of the application database. Django officially supports four application databases: PostgreSQL, MySQL, SQLite, and Oracle.

Model in Django is a special Python class and it holds business logic, properties, specific methods, and other behavior of manipulated data. Models in Django allow developers to create, read, update, and delete data in the original database without using complicated SQL expressions. Django also automatically generates web pages for maintaining model data on special administrative site of Django application [12].

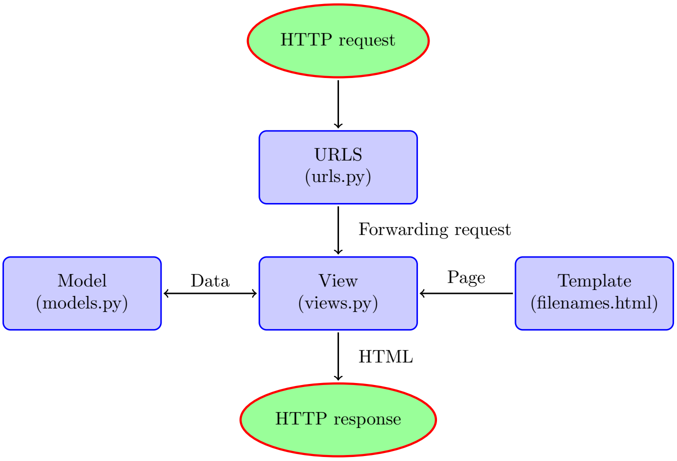

Figure 2. Django Model-View-Template architecture

```
class OR_Business_Process2(models.Model):
    OR_business_process2_ID = models.CharField(max_length=2, unique = True,
        verbose_name = "Business Process Code")
    OR_business_process_ID = models.ForeignKey(OR_Business_Process, on_delete=
        models.DO_NOTHING, verbose_name = "Business Line Code")
    OR_business_process2_name = models.CharField(max_length=50,
        verbose_name = "Business Process Name")
    OR_business_process2_description = models.TextField(max_length=500,
        verbose_name = "Business Process Description")
    class Meta:
        ordering = ["OR_business_process2_ID"]
        verbose_name = "Business Process"
        verbose_name_plural = "Business Processes"
    def __str__(self):
        """String for representing the OR_Business_Process2 object"""
        return '%s) %s' % (self.OR_business_process2_ID,
            self.OR_business_process2_name)
```

Figure 3. Django model class for business process classifier

An example of model for business process (level 2) classifier is given in Fig. 3.

The view component of Django executes three main tasks: it accepts HTTP requests from users, applies business logic from Django models, and provides HTTP responses to users' requests.

Django has a powerful template engine and its own markup language with many tools. Templates in Django are HTML files used to present data.

Django has built-in object-relational mapping (ORM) that helps developers interact with databases. ORM is a mechanism that automatically transfers data stored in application database into objects used in application code [13].

Django ORM speeds up web application development and helps developers build working prototypes in reduced time. Developers may not know the details of SQL implementation for a specific database to manipulate data.

The Django framework can be used for creating various applications, including client relationship management (CRM) systems, communication platforms, booking engines, machine learning systems and many other applications [14, 15].

Due to advantages of Django architecture operational risk event database application can be rapidly implemented in Django web framework.

4. Value-at-Risk and Expected Shortfall Calculation

Operational risk may be measured using various approaches and risk metrics, for instance, Basel II approaches (basic indicator approach, standardized approach, advanced measurement approaches), expected loss and unexpected loss, etc [16].

Value-at-Risk (VaR) and Expected Shortfall (ES) are the most widely used financial risk metrics [17, 18]. In market risk management Value-at-Risk is the worst loss of asset portfolio over a given time interval that will not be exceeded with a given confidence level. Expected Shortfall at a given confidence level α is the expected loss of asset portfolio in the worst $1-\alpha$ percent of losses.

Value-at-Risk and Expected Shortfall can be calculated for operational risk in a number of ways [19]. If event database with losses incurred due to operational risk events is available for certain time period, then the calculation of VaR and ES may be based on data in risk event database [20].

Assume that operational risk loss L per day is distributed discretely, i.e. operational risk loss per day L for the period of n days takes some values l_1, l_2, ..., l_n with equal probability $p = \frac{1}{n}$, i.e.

$$\mathbb{P}\left[L = l_i\right] = p = \frac{1}{n},\ i = \overline{1, n}.$$

Suppose for simplicity that $l_i \neq l_j$, $i \neq j$ and day losses l_i are sorted in ascending order:

$$0 < l_1 < l_2 < ... < l_n.$$

Due to property $\mathbb{P}\left[L \leqslant l\right] = 1 - \mathbb{P}\left[L > l\right]$ the definition of $VaR\,(\alpha)$ takes the form

$$VaR\,(\alpha) = \inf\{l \in \mathbb{R} : \mathbb{P}\left[L \leqslant l\right] \geqslant \alpha\} = \inf\{l \in \mathbb{R} : \mathbb{P}\left[L > l\right] \leqslant 1 - \alpha\}.$$

If $l \in [l_k, l_{k+1})$ for some index k between 1 and n $(1 \leqslant k < n)$, then

$$\mathbb{P}\left[L > l\right] = \sum_{i=k+1}^{n} \mathbb{P}\left[L = l_i\right] = \frac{n-k}{n}.$$

If for some k the confidence level α is equal to

$$\alpha = \frac{k}{n},\ 1 \leqslant k < n,\ k \in \mathbb{N},$$

then for any $l \in \mathbb{R}$ inequality

$$\mathbb{P}\left[L > l\right] \leqslant 1 - \alpha = 1 - \frac{k}{n} = \frac{n-k}{n}$$

is equivalent to inequality $l \geqslant l_k$, therefore

$$VaR\,(\alpha) = \inf\{l \in \mathbb{R} : \mathbb{P}\left[L > l\right] \leqslant 1 - \alpha\} = \inf\{l \in \mathbb{R} : l \geqslant l_k\} = l_k.$$

Algorithm: calculation of operational VaR and ES on a given event database.

Input:

```
1. start date $t_1$ and end date $t_2$ of time interval $[t_1, t_2]$
2. event database with operational loss data for time interval $[t_1, t_2]$
3. confidence level $\alpha \in (0, 1)$
```

Output: daily operational VaR and ES with confidence level α

Method:

```
(1)  calculate total number of days in time interval $[t_1, t_2]$: $N = \frac{t_2 - t_1 + 1}{365}$
(2)  for each day of time interval $[t_1, t_2]$ sum operational losses incurred on that day
(3)  sort the losses received on previous step in descending order and store the losses in array $A$ (days with high losses at the beginning of array, array index starts from 1)
(4)  if (confidence level $\alpha$ is equal to $\frac{k}{N}$ for some integer $k$ between 1 and $N$) {
(5)      $VaR(\alpha) = A[N - k + 1]$
(6)      $ES(\alpha) = \frac{1}{N-k} \sum_{i=1}^{N-k} A[i]$
(7)  }
(8)  else if ($\frac{k}{n} < \alpha < \frac{k+1}{n}$ for some integer $k$ between 1 and $N - 1$) {
(9)      $VaR(\alpha) = A[N - k]$
(10)     $ES(\alpha) = \frac{1}{1-\alpha} \left( \frac{1}{N} \sum_{i=1}^{N-k-1} A[i] + \left( \frac{k+1}{n} - \alpha \right) A[N-k] \right)$
(11) }
```

Figure 4. Algorithm for operational VaR and ES calculation

If for some k the confidence level α satisfies the double inequality

$$\frac{k}{n} < \alpha < \frac{k+1}{n},\ 1 \leqslant k < n,\ k \in \mathbb{N},$$

then for any $l \in [l_k, l_{k+1})$

$$\mathbb{P}[L > l] = \frac{n-k}{n} > 1 - \alpha,$$

hence $\mathbb{P}[L > l] < 1 - \alpha$ only when $l \geqslant l_{k+1}$ and

$$VaR(\alpha) = \inf\{l \in \mathbb{R} : l \geqslant l_{k+1}\} = l_{k+1}.$$

So finally under made assumptions

$$VaR(\alpha) = \begin{cases} l_k, & \text{when } \alpha = \frac{k}{n}, \\ l_{k+1}, & \text{when } \frac{k}{n} < \alpha < \frac{k+1}{n}, \end{cases}$$

where $1 \leqslant k < n$, $k \in \mathbb{N}$.

For the calculation of Expected Shortfall for given confidence level α we have to determine worst $1 - \alpha$ percent of losses and calculate average value for these losses.

For $\alpha = \frac{k}{n}$, $1 \leqslant k < n$, $k \in \mathbb{N}$ we have $1 - \alpha = \frac{n-k}{n}$, so worst $1 - \alpha$ percent of losses are the last $n - k$ losses $l_{k+1}, ..., l_n$ and

$$ES\,(\alpha) = \mathbb{E}\,[L \mid L > l_k] = \frac{\sum_{i=k+1}^{n} p\, l_i}{\sum_{i=k+1}^{n} p} = \frac{1}{n-k} \sum_{i=k+1}^{n} l_i.$$

For $\frac{k}{n} < \alpha < \frac{k+1}{n}$, $1 \leqslant k < n$, $k \in \mathbb{N}$ we have $1 - \alpha = \tilde{p} + \frac{n-k-1}{n}$ for some $\tilde{p} \in \left(0,\, \frac{1}{n}\right)$, $\tilde{p} = 1 - \alpha - \frac{n-k}{n}$, so worst $1 - \alpha$ percent of losses are the last $n - k - 1$ losses $l_{k+2}, ..., l_n$ and a part of the loss l_{k+1} and

$$ES\,(\alpha) = \frac{\tilde{p}\, l_{k+1} + \sum_{i=k+2}^{n} p\, l_i}{\tilde{p} + \sum_{i=k+2}^{n} p} = \frac{1}{1-\alpha} \left[\tilde{p}\, l_{k+1} + \frac{1}{n} \sum_{i=k+2}^{n} l_i\right]$$

So under made assumptions

$$ES\,(\alpha) = \begin{cases} \frac{1}{n-k} \sum_{i=k+1}^{n} l_i, & \text{when } \alpha = \frac{k}{n}, \\ \frac{1}{1-\alpha} \left[\tilde{p}\, l_{k+1} + \frac{1}{n} \sum_{i=k+2}^{n} l_i\right], & \text{when } \frac{k}{n} < \alpha < \frac{k+1}{n}, \end{cases}$$

where $1 \leqslant k < n$, $k \in \mathbb{N}$.

From received formulae for $VaR\,(\alpha)$ and $ES\,(\alpha)$ we receive the following algorithm (Fig. 4) for the calculation of Value-at-Risk and Expected Shortfall under made assumptions.

Algorithm on Fig. 4 can be easily implemented in Python, using methods `aggregate` and `annotate` of Django database API.

5. Conclusion

We presented some design and implementation considerations for a web based application for operational loss collection. Using Django web framework significantly speeds up and facilitates application development. Algorithm for calculation of operational Value-at-Risk and Expected Shortfall using data in risk event database is also developed.

Acknowledgments

The publication has been prepared with the support of the «RUDN University Program 5-100» (risk event database design and implementation). The reported study was funded by RFBR, grant No 16-08-00558 (VaR and ES algorithm development).

References

1. Bank of Russia, On the requirements to the risk and capital management system of credit institution and banking group (ordinance no. 3624-u of april 15, 2015), Vestnik Banka Rossii (51) (2015) 15–35.
2. A. Aloqab, F. Alobaidi, B. Raweh, Operational risk management in financial institutions: An Overview, Business and Economic Research 8 (2) (2018) 11. `doi:10.5296/ber.v8i2.12681`.
3. S. Pakhchanyan, Operational risk management in financial institutions: A Literature Review, International Journal of Financial Studies 4 (4) (2016) 20. `doi:10.3390/ijfs4040020`.

4. BCBS, Basel III: Finalising post-crisis reforms, Publication, Basel Committee on Banking Supervision, Basel (dec 2017).
URL `http://www.bis.org/bcbs/publ/d424.pdf`
5. Bank of Russia, On the requirements to the operational risk management system of credit institution and banking group (draft regulation), Draft regulation, Moscow (September 2018).
URL `http://www.cbr.ru/StaticHtml/File/41186/180918-41_1.pdf`
6. L. Wei, J. Li, X. Zhu, Operational loss data collection: A Literature Review, Annals of Data Science 5 (3) (2018) 313–337. `doi:10.1007/s40745-018-0139-2`.
7. BCBS, The Quantitative Impact Study for Operational Risk: Overview of Individual Loss Data and Lessons Learned, Working paper, Basel Committee on Banking Supervision, Basel (jan 2002).
URL `https://www.bis.org/bcbs/qis/qisopriskresponse.pdf`
8. D. Beazley, G. Van Rossum, Python: Essential Reference, New Riders Publishing, 1999.
9. A. Holovaty, J. Kaplan-Moss, The Definitive Guide to Django, Apress, 2009. `doi:10.1007/978-1-4302-1937-8`.
10. A. Mével, T. Guéguen, M. Wolczko, Smalltalk-80, Macmillan Education UK, 1987. `doi:10.1007/978-1-349-09653-4`.
11. G. E. Krasner, S. T. Pope, A cookbook for using the model-view controller user interface paradigm in smalltalk-80, Journal of Object-Oriented Programming 1 (3) (1988) 26–49.
URL `http://dl.acm.org/citation.cfm?id=50757.50759`
12. M. Alchin, Pro Django, Apress, 2013. `doi:10.1007/978-1-4302-5810-0`.
13. M. Lorenz., G. Hesse., J. Rudolph., Object-relational mapping revised - a guideline review and consolidation, in: Proceedings of the 11th International Joint Conference on Software Technologies - Volume 1: ICSOFT-EA, (ICSOFT 2016), INSTICC, SciTePress, 2016, pp. 157–168. `doi:10.5220/0005974201570168`.
14. K. Cao, F. Wang, J. G. Liu, Study and implementation of pm2.5 data download service based on python, Applied Mechanics and Materials 411-414 (2013) 555–558. `doi:10.4028/www.scientific.net/amm.411-414.555`.
15. Z. Li, Design and implementation of the software testing management system based on django, Applied Mechanics and Materials 525 (2014) 707–710. `doi:10.4028/www.scientific.net/amm.525.707`.
16. R. Coleman, A VaR too far? The pricing of operational risk, Journal of Financial Transformation 28 (2010) 123–129.
URL `https://EconPapers.repec.org/RePEc:ris:jofitr:1420`
17. P. Jorion, Value at Risk: The New Benchmark for Managing Financial Risk, 3rd Edition, Mcgraw-Hill Professional, 2006.
18. C. Acerbi, D. Tasche, Expected shortfall: A Natural Coherent Alternative to Value at Risk, Economic Notes 31 (2) (2002) 379–388. `doi:10.1111/1468-0300.00091`.
19. A. S. Chernobai, S. T. Rachev, F. J. Fabozzi, Operational Risk: A Guide to Basel II Capital Requirements, Models, and Analysis, John Wiley & Sons, Inc., 2012. `doi:10.1002/9781119201922`.
20. J. Esterhuysen, P. Styger, G. W. V. Vuuren, Calculating operational value-at-risk (opvar) in a retail bank, South African Journal of Economic and Management Sciences 11 (1) (2012) 1–16. `doi:10.4102/sajems.v11i1.374`.