

STRESS TESTING

FIELD OF THE INVENTION

5 The present invention relates to risk management in financial institutions. More particularly, the invention relates to system and method for stress testing in enterprises for risk management.

BACKGROUND

10

Financial institutions strive to establish a risk management framework to avoid incidents. Enterprise Stress Testing (EST) for long has been a crucial component of the framework that has continuously evolved across sectors of the financial services industry including governance, modeling, data management and optimization for
15 avoiding incidents. EST solutions have to hold multiple technologies covering performance and data. The EST solutions help estimate the likely losses that a financial institution may suffer under exceptional scenarios.

Techniques used to comply with requirements of determination of EST vary with
20 different sectors. For a bank, stress-testing technology can cost a huge amount every year, but despite the costs involved the banks are putting a lot of effort, time and money in trying to invest in sustainable processes.

Data management is critical to success of EST systems as overall stress testing is
25 essential to the success of all aspects of financial services operations. Risk-based accounting management that demands changes to financial institutions processes and operations is another external factor that shapes growing need for EST.

Financial institutions need to ensure that they have appropriate performance indicator
30 and create enterprise-wide risk environment to support their business requirements. The mix-match approach followed in general is not viable for accurate determination of enterprise stress.

Further, the existing systems work in silos. Hence, for upper management enterprise
35 level if viewing is not available, it certainly means that the management lacks

perspective on the impact of scenario created by one department to other department, and lacks perspective on the impact of one common scenario. EST should impact business decisions at a high level and on a day-to-day basis, instead of being an isolated compliance activity. In order to remove silos and ensure a strategic approach, 5 disparate processes and structures need to be merged into an end-to-end stress testing process that is unified and addresses impacts of multiple factors. However, the merging of distinct processes requires underlining changes to the existing structures which are not feasible considering the fact that the data parameters and risk factors keep changing dynamically. If the underlining structure is changed every 10 time a new process or data is included, it would make the system error-prone, more time consuming due to slow processing and would require additional memory for storing modified protocols.

In view of the above, there exists a need of improved systems and methods that 15 overcome the shortcomings associated with existing technologies and prior arts.

SUMMARY OF THE INVENTION

Accordingly, the present invention provides a method for stress testing. The method 20 comprises the steps of receiving by a processor at least one risk factor to be associated with an enterprise framework. The method includes storing by seeding the risk factor in a risk factor database through a database initializer wherein a seed data model of the enterprise framework is configured to create an underlining seeded mapping structure for the risk factor, identifying at least one key risk factor from the 25 database based on analysis of a plurality of risk factors associated with the enterprise framework, associating an impact parameter with the identified key risk factor. The method also includes generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter, and triggering at least one risk engine associated with the at least one stress scenario and the identified 30 key risk factor for determining stress wherein the risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value.

In an embodiment, the present invention provides a system for enterprise stress 35 testing. The system includes an electronic user interface configured for operating on

an enterprise stress testing application, at least one risk factor database configured for storing a plurality of risk factors wherein a new risk factor to be associated with an enterprise framework is stored by seeding the risk factor in the database through a database initializer wherein a seed data model of the enterprise framework is configured to create an underlining seeded mapping structure for the new risk factor. The system includes a controller encoded with instructions enabling the controller to function as a bot wherein the controller is configured to perform, identifying at least one key risk factor from the database based on analysis of the plurality of risk factors associated with the enterprise framework; associating an impact parameter with the identified key risk factor; generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter. The system includes a processor coupled to the controller and configured for triggering at least one risk engine associated with the at least one stress scenario and the identified key risk factor for determining stress wherein the risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value.

In an embodiment, the present invention provides a computer-readable non-transitory storage medium storing executable program instructions for enterprise stress testing which when executed by a computer causes the computer to perform operations as described above.

In an advantageous aspect, the user interface and underlying logic of the stress testing is based the seeded mapping. Seeded mappings explicitly refer to the list of risk factor(s) that are configured in the system. Every entry of seeded risk factor has its corresponding dataset joins which are also seeded. These joins are read by the underlying code which creates the dynamic query to fetch the data needed to define the stress scenario on the risk factor. If there are new risk factor that are introduced, then system will automatically include the risk factor in user interface as well as the code itself will create the underlying code for new risk factor based on the seeded entries of the risk factor. No code change will be required for doing so. As seeded entries are combination of risk factor and database joins that are needed to create the stress scenario hence no code changes are needed. In absence of the seeded entries, the entire logic of capturing the data required to for creating stress scenario will be hardcoded within the code, instead underlying code reads the dataset joins

from seeded entries and allows system to create the dynamic queries which helps stress scenario to fetch the relevant data. If seeded entries are not available, then code change of fetching the relevant data for stress scenario will have to be done for every new risk factor that will be included within EST framework.

5

In another advantageous aspect, all components which are required while interacting with the various engines are auto-built while creating the scenario, as building of these components are also based on the underlying configuration. Every risk factor has a unique feature of type of shocks that can be applied to it based on its usage within the application. For example, all quantitative risk factor can have percentage or absolute shock, and qualitative risk factor can have relative or rank shock. Underlying configuration enables user to map this shock type to the list of risk factors. EST application develops the json component which helps engines to understand the nature of the shock type and perform the required computation. Along with the generic component specified, EST also builds the application specific component which captures bare minimum attributes that are needed to execute the engine appropriately. Once the stress scenario is triggered, the system initiates interaction with all engines, understands the sequence in which engines should be triggered, captures the stress output from all the engines, consolidate output across engines and provide one enterprise level stress view across engines thereby enabling management to take appropriate action.

Additionally, the system application also creates the dependency matrix of risk engines. Dependency matrix gives EST application framework capability to trigger a plurality of risk factor engines simultaneously, thus saving processing time as compared to sequential execution of the risk engines. The process of creating the sequence in which engines are supposed to be triggered is called as creation of dependencies matrix of risk engines. Dependencies matrix is determined by application by analyzing the way the stress scenario is configured by user. System executes the engines sequentially or simultaneously based in the dependencies matrix. For example, the system executes the engine in a sequential manner if it detects that an input of an engine, for example engine2, is an output of another engine, for example of engine1. In such a scenario, the system will execute engine sequentially, first it will execute engine1 and then execute engine2. In the scenario where system does not detect such dependencies, i.e., inputs to an engine are

independent to the output of another engine, the system will trigger such engines simultaneously.

DESCRIPTION OF THE DRAWINGS

5

Fig. 1 shows a system architecture for stress testing in accordance with an embodiment of the present invention.

Fig. 2 shows a functional flow diagram for a stress testing enterprise framework in accordance with an embodiment of the present invention.

10

Fig. 3 shows a flow diagram depicting a method of stress testing in accordance with an embodiment of the present invention.

15

DESCRIPTION OF THE INVENTION

Various embodiments of the present invention provide systems and methods for stress testing by enterprise framework in risk environment. The following description provides specific details of certain embodiments of the invention illustrated in the drawings to provide a thorough understanding of those embodiments. It should be recognized, however, that the present invention can be reflected in additional embodiments and the invention may be practiced without some of the details in the following description.

20

The various embodiments including the example embodiments will now be described more fully with reference to the accompanying drawings, in which the various embodiments of the invention are shown. The invention may, however, be embodied in different forms and should not be construed as limited to the embodiments set forth herein. Rather, these embodiments are provided so that this disclosure is thorough and complete, and fully conveys the scope of the invention to those skilled in the art. In the drawings, the sizes of components may be exaggerated for clarity.

30

It will be understood that when an element or layer is referred to as being “on” “connected to” or “coupled to” another element or layer, it can be directly on, connected to, or coupled to the other element or layer or intervening elements or

35

layers that may be present. As used herein, the term “and/or” includes any and all combinations of one or more of the associated listed items.

Spatially relative terms, such as “scenario,” “risk factor,” “financial institutions” and the like, may be used herein for ease of description to describe one element or feature's relationship to another element(s) or feature(s) as illustrated in the figures. It should be understood that the spatially relative terms are intended to encompass different orientations of the structure in use or operation in addition to the orientation depicted in the figures.

Embodiments described herein will refer to plan views and/or cross-sectional views by way of ideal schematic views. Accordingly, the views may be modified depending on simplistic assembling or manufacturing technologies and/or tolerances. Therefore, example embodiments are not limited to those shown in the views but include modifications in configurations formed on basis of assembling process. Therefore, regions or regions of elements exemplified in the figures have schematic properties and shapes, and do not limit the various embodiments including the example embodiments.

The subject matter of example embodiments, as disclosed herein, is described with specificity to meet statutory requirements. However, the description itself is not intended to limit the scope of this patent. Rather, the inventors have contemplated that the claimed subject matter might also be embodied in other ways, to include different features or combinations of features similar to the ones described in this document, in conjunction with other technologies. Generally, the various embodiments including the example embodiments relate to system and method for stress testing in risk environments.

Referring to Fig. 1, a system architecture 100 for a stress testing enterprise framework is shown in accordance with an embodiment of the present invention. The system 100 include at least one computing device 110, a server support architecture 120, a data processing and control support architecture/mechanism 130, a data storage support architecture 140, an underlining structure modification support interface 150, and a scheduler 160. The server support architecture may include server 120a and mainframe 120b. The data processing and control support architecture/mechanism

130 may include a processor 130a, a controller 130b, a database initializer 130c and a data mapper 130d. The data storage support architecture 140 may include a data lake 140a, a risk factor database 140b and a data model database 140c.

5 Referring to Fig. 2, a functional process flow diagram 200 of the stress testing enterprise framework is shown in accordance with an embodiment of the present invention. The flow diagram includes various constituting elements of the framework including but not limited to input components 210 such as account and instrument details from universal data lake, risk factors like credit risk, macroeconomic risk,
10 operational risk, liquidity risk, financial risk etc., analysis attributes such as currency, branch, business segment, country, customer type, industry, legal entity, product etc.; the plurality of engines 220 such as credit risk engine, liquidity risk engine, financial ratio engine, cashflow and valuation engine, market risk engine, ECL (Expected Credit Loss) engine; stress scenario components 230 such as banks custom scenario, Japanese financial system, Asian financial crises etc; a simulation engine 240 for
15 simulating the various scenarios and reports component 250 that includes overview of the baseline, stress scenario and worst scenario.

In an embodiment, once the stress scenario is created it needs to be executed on
20 predetermined frequency to analyze the results. This is achieved without any manual intervention application, and this creates the batches. Scheduler then triggers the batches based on the frequency configured within the scheduler, hence provides the regular update of configured stress scenario.

25 Referring to Fig. 1 and 2, the system 100 includes an electronic user interface configured for operating on an enterprise stress testing application, at least one risk factor database 140b configured for storing a plurality of risk factors wherein a new risk factor to be associated with an enterprise framework is stored by seeding the risk factor in the database 140b through the database initializer 130c wherein a seed data
30 model in the data model database 140c of the enterprise framework is configured to create an underlining seeded mapping structure for the new risk factor. Seeded mappings explicitly refer to the list of risk factor(s) that are configured in the system. Every entry of seeded risk factor has its corresponding dataset joins which are also seeded. These joins are read by the underlying code which creates the dynamic query
35 to fetch the data needed to define the stress scenario on the risk factor.

In an embodiment the controller 130b of the system 100 is encoded with instructions enabling the controller 130b to function as a bot wherein the controller 130b is configured to perform the functions of, identifying at least one key risk factor from the database 140b based on analysis of the plurality of risk factors associated with the enterprise framework, associating an impact parameter with the identified key risk factor, generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter. The processor 130a coupled to the controller 130b is configured for triggering at least one risk engine associated with the at least one stress scenario and the identified key risk factor for determining stress. The risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value.

In an example embodiment the server 120a may include electronic circuitry for enabling execution of various steps by the processor. The electronic circuitry may have various elements including but not limited to a plurality of arithmetic logic units (ALU) and floating-point Units (FPU), and/or the equivalents thereof. The ALU enables processing of binary integers to assist in seeded mapping where a seed data model in the data model database 140c of the enterprise framework is configured to create an underlining seeded mapping structure for the new risk factor. In an example embodiment, the server electronic circuitry includes at least one arithmetic logic unit, floating point units (FPU), other processors, memory, storage devices, high-speed interfaces connected through buses for connecting to memory and high-speed expansion ports, and a low speed interface connecting to low speed bus and storage device. The components of the electronic circuitry are interconnected using various busses, and may be mounted on a common motherboard or in other manners as appropriate. The processor can process instructions for execution within the server 120a, including instructions stored in the memory or on the storage devices to display graphical information for a GUI on an external input/output device, such as display coupled to high speed interface. In other implementations, multiple processors and/or multiple busses may be used, as appropriate, along with multiple memories and types of memory. Also, multiple servers may be connected with each server providing portions of the necessary operations (e.g., as a server bank, a group of blade servers, or a multi-processor system).

35

In an embodiment, the risk engine is an artificial intelligence-based engine configured for processing a plurality of data models associated with the stress scenario for determining the enterprise stress.

5 The processor 130a may communicate with a user through a control interface and display interface coupled to a display. The display may be, for example, a TFT LCD (Thin-Film-Transistor Liquid Crystal Display) or an OLED (Organic Light Emitting Diode) display, or other appropriate display technology. The display interface may comprise appropriate circuitry for driving the display to present graphical and other
10 information to an entity/user. The control interface may receive commands from a user and convert them for submission to the processor. In addition, an external interface may be provided in communication with processor 130a so as to enable near area communication of device with other devices. External interface may be suitable, for example, for wired communication in some implementations, or for wireless
15 communication in other implementations, and multiple interfaces may also be used.

The data storage support architecture 140 may include memory units that may be a volatile, a non-volatile memory or memory may also be another form of computer-readable medium, such as a magnetic or optical disk.

20 The data storage 140 may also include storage device capable of providing mass storage. In one implementation, the storage device may be or contain a computer-readable medium, such as a floppy disk device, a hard disk device, an optical disk device, a tape device, a flash memory or other similar solid state memory device, or
25 an array of devices, including devices in a storage area network or other configurations.

In an embodiment the risk factor includes credit rating, market rating, liquidity, operational, Probability of Default (PD), Loss Given Default (LGD), and Exposure at
30 Default (EAD).

In an embodiment, the risk engine is configured for reading a base data of the at least one identified risk factor from the database.

In an embodiment, the engine includes Risk Weighted Asset (RWA) engine, market risk engine, financial ratio risk, ECL risk, liquidity risk.

5 In an embodiment, the controller is configured for creating an enterprise level stress scenario from a plurality of stress scenarios wherein the enterprise level stress scenario is applied to a plurality of identified risk engines of the enterprise framework for determining an enterprise level stress.

10 In an embodiment, the impact parameter is associated with the identified key risk factor based on an input received from a historical database or a real time user input or a combination of both.

15 In an embodiment, the present invention includes a natural language processing (NLP) server configured for processing the stress scenario based on the plurality of data models.

20 In an embodiment, the user interface and an underlying logic of the stress testing is based on the seeded mapping wherein on introduction of the new risk factor the system automatically includes the risk factor in the user interface wherein the underlying seeded mapping structure for the new risk factor is automatically created based on seeded data entries of the risk factor.

25 The enterprise framework functions on seeded data and generates mapping structures or codes on basis of the seeded data thereby enabling inclusion of a new engine or the new risk factor within the framework upon updating of seeded entries.

30 In an embodiment, the present invention provides a dependency data matrix of the plurality of risk engine configured for triggering the plurality of risk engines sequentially or parallelly depending on a dynamic processing logic.

35 In an exemplary embodiment of the present invention, a plurality of components required for interacting with the plurality of identified risk engines are auto-built while creating the stress scenario, as building of these components are based on an underlying configuration of the system wherein once the stress scenario is triggered, the controller coupled to the processor interacts with the plurality of engines,

determines a sequence of triggering the engines based on the dynamic processing logic, captures a stress value each from the plurality of engines, consolidates each of the stress value and provides one enterprise level stress value thereby enabling management to take appropriate action.

5

In an embodiment, the present invention provides a non-transitory storage medium storing executable program instructions for enterprise stress testing which when executed by a computer causes the computer to perform stress testing operations.

10 In an embodiment, the present invention provides executable program instructions in a memory to be executed for generating a plurality of data models.

Referring to Fig. 3, a flowchart 300 depicting a method for stress testing is provided in accordance with an embodiment of the present invention. The method comprises
15 the steps of (S310) receiving by a processor at least one risk factor to be associated with an enterprise framework; (S320) storing by seeding the risk factor in a risk factor database through a database initializer wherein a seed data model of the enterprise framework is configured to create an underlining seeded mapping structure for the risk factor; (S330) identifying at least one key risk factor from the database based on
20 analysis of a plurality of risk factors associated with the enterprise framework; (S340) associating an impact parameter with the identified key risk factor; (S350) generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter; (S360) triggering at least one risk engine associated with the at least one stress scenario and the identified key risk factor for determining
25 stress wherein the risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value.

In some implementations, the machine learning data model is configured and trained
30 to map underlying structure of only a risk factor/object. For example, output generated over the model may provide an indication of whether a particular object or class of objects is present, and optionally user instructions. In some implementations, the machine learning model is configured and trained to map underlining structures of multiple risk factors/objects. Accordingly, in those implementations a single pass over
35 a single machine learning model may be utilized to detect whether each of multiple

objects is present. For example, output generated over the model may provide an indication of whether a first risk factor/object is present, and indication of whether a second particular risk factor/object or class of object is present, etc.—and optionally performance for one or more of the particular risk factors indicated to be present.

5

It should be apparent that different aspects of the description provided above may be implemented in many different forms of software, firmware, and hardware in the implementations illustrated in the figures. The actual software code or specialized control hardware used to implement these aspects is not limiting of the invention.

10 Thus, the operation and behavior of these aspects were described without reference to the specific software code—it being understood that software and control hardware can be designed to implement these aspects based on the description herein.

In one example embodiment, the stress testing is performed for a stressed RWA, key risk factor for stressing RWA will be credit rating, PD, LGD and EAD. Once key risk factors are identified, they are associated to the stress scenario and an impact parameter like a shock value will be associated with them. This shock value can be based on expert's judgment as well as based on historical event. Various types of shock that can be provided for the risk factors are: (1) Percentage Shock: this is applicable to all Quantitative risk factor, (2) Absolute Shock: this is applicable to all Quantitative risk factor, (3) Rank Shock: this is applicable to all Qualitative risk factor which have values in the form of relative ranking/Grade like credit rating (A, AA, AAA), (4) Simulation based: this is mainly applicable to risk factor whose worst case value are to be statistically predicted using either independent simulation or joint simulation, (5) Equation Based Shock: this will be applicable to the risk factors which are dependent on other risk factor and user is aware of the relationship/equation between the stated risk factors, (6) Engine Based: this is applicable to the risk factor which are output of one engine and input of another engine and (7) Categorical Risk Factor: this will be applicable to the Qualitative risk factor which are categorical in nature, for example, geographical sector (APAC - Asia Pacific, EMEA - Europe, Middle-East and Africa, AMER - North, Central and South America).

In a related example embodiment, the system based on historical data has generated created scenario with only one key risk factor PD. Within scenario the historical data provided that PD should be shocked by 10%, hence input to the stress scenario is

35

selection of risk factor PD and shock value of 10%. Once scenario is generated, the user executes from stress scenario user interface (or this can also be converted into a batch and can be triggered from scheduler) to get the stress result. Internally, EST will trigger the RWA engine and pass shock value of PD (i.e., 10% in given example) to RWA engine. RWA engine will read the base data of all the required risk factor from data base and apply the shock value received from stress scenario. For given example embodiment, RWA engine will read base value of Credit Rating, PD, LGD, EAD and other inputs that are required by RWA engine. RWA engine has also received the information from EST about the shock value of PD (i.e., 10%). RWA engine will update the base value of PD (i.e., increase it PD by 10%) and compute the RWA based on shocked value of PD to generate shocked RWA value.

It will be understood to a person skilled in the art that in the above example only one risk factor PD is shocked. The stress scenario can be generated to shock multiple risk factor in the same scenario. So, if the scenario is generated by shocking both the PD and PGD, then RWA engine will update the base value of both risk factor PD and LGD and determine the shocked RWA value. In the above example, the input and output are given from RWA engine point of view. Similarly, shocked values can be passed to other engines like "Market Risk", "Financial Ratio", "ECL", "Liquidity Risk" and other engines that will be integrated with EST application. This provides the user benefit to maintain the stress scenario at the single place and also provide ability to create "Enterprise level Stress Scenario" and apply the same scenario to all risk engines.

Further, certain portions of the invention may be implemented as a "component" or "system" that performs one or more functions. These components/systems may include hardware, such as a processor, an ASIC (Application Specific Integrated Circuit), or a FPGA (Field Programmable Gate Array), or a combination of hardware and software.

The word "exemplary" is used herein to mean "serving as an example." Any embodiment or implementation described as "exemplary" is not necessarily to be construed as preferred or advantageous over other embodiments or implementations.

No element, act, or instruction used in the present application should be construed as critical or essential to the invention unless explicitly described as such. Also, as used herein, the article “a” and “one of” is intended to include one or more items. Further, the phrase “based on” is intended to mean “based, at least in part, on” unless explicitly
5 stated otherwise.

Each of the above identified processes corresponds to a set of instructions for performing a function described above. The above identified programs or sets of instructions need not be implemented as separate software programs, procedures or
10 modules, and thus various subsets of these modules may be combined or otherwise re-arranged in various embodiments. For example, embodiments may be constructed in which steps are performed in an order different than illustrated, steps are combined, or steps are performed simultaneously, even though shown as sequential steps in illustrative embodiments. Also, the terminology used herein is for the purpose of
15 description and should not be regarded as limiting. The use of “including,” “comprising,” “having,” “containing” and variations thereof herein, is meant to encompass the items listed thereafter and equivalents thereof as well as additional items.

20 The above-described embodiments of the present invention may be implemented in any of numerous ways. For example, the embodiments may be implemented using various combinations of hardware and software and communication protocol(s). Any standard communication or network protocol may be used and more than one protocol may be utilized. For the portion implemented in software, the software code may be
25 executed on any suitable processor or collection of processors, whether provided in a single computer or distributed among multiple computers. Such processors may be implemented as integrated circuits, with one or more processors in an integrated circuit component, or any other suitable circuitry. Further, it should be appreciated that a computer may be embodied in any of a number of forms, such as a rack-
30 mounted computer, a desktop computer, a laptop computer, single board computer, micro-computer, or a tablet computer. Additionally, a computer may be embedded in a device not generally regarded as a computer but with suitable processing capabilities, including a Personal Digital Assistant (PDA), a smart phone or any other suitable portable or fixed electronic device.

35

Also, the various methods or processes outlined herein may be coded as software that is executable on one or more processors that employ any one of a variety of operating systems or platforms. Additionally, such software may be written using any of a number of suitable programming languages and/or programming or scripting tools
5 or a combination of programming languages, and also may be compiled as executable machine language code or intermediate code that is executed on a framework or a virtual machine. In this respect, the invention may be embodied as a computer readable storage medium (or multiple computer readable media) (e.g., a computer memory, one or more floppy discs, compact discs (CD), optical discs, digital video
10 disks (DVD), magnetic tapes, flash memories, circuit configurations in Field Programmable Gate Arrays or other semiconductor devices, or other tangible computer storage medium) encoded with one or more programs that, when executed on one or more computers or other processors, perform methods that implement the various embodiments of the invention discussed above. As is apparent from the
15 foregoing examples, a computer readable storage medium may retain information for a sufficient time to provide computer-executable instructions in a non-transitory form.

The terms "program" or "software" are used herein in a generic sense to refer to any type of computer code or set of computer-executable instructions that may be
20 employed to program a computer or other processor to implement various aspects of the present invention as discussed above. Additionally, it should be appreciated that according to one aspect of this embodiment, one or more computer programs that when executed perform methods of the present invention need not reside on a single computer or processor but may be distributed in a modular fashion amongst a number
25 of different computers or processors to implement various aspects of the present invention. Computer-executable instructions may be in many forms, such as program modules, executed by one or more computers or other devices. Also, data structures may be stored in computer-readable media in any suitable form. Any suitable mechanism may be used to establish a relationship between information in fields of a
30 data structure, including the use of pointers, tags, or other mechanisms that establish relationship between data elements.

It is to be understood that the above-described embodiments are only illustrative of the application of the principles of the present invention. The illustrative discussions
35 above are not intended to be exhaustive or to limit the invention to the precise forms

disclosed. Various modifications and alternative applications may be devised by those skilled in the art in view of the above teachings and without departing from the spirit and scope of the present invention and the following claims are intended to cover such modifications, applications, and embodiments.

5

CLAIMS

1. A method for stress testing, the method comprises the steps of:
 - receiving by a processor at least one risk factor to be associated with an enterprise framework;
 - storing by seeding the risk factor in a risk factor database through a database initializer wherein a seed data model of the enterprise framework is configured to create an underlining seeded mapping structure for the risk factor;
 - identifying at least one key risk factor from the database based on analysis of a plurality of risk factors associated with the enterprise framework;
 - associating an impact parameter with the identified key risk factor;
 - generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter wherein a dataset join corresponding to the identified key risk factor is read by an underlying code that creates a dynamic query to fetch data needed to define the stress scenario on the risk factor; and
 - triggering at least one risk engine associated with the at least one stress scenario and the identified key risk factor for determining stress wherein the risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value;
 - wherein a user interface and an underlying logic of the stress testing is based on the seeded mapping wherein on introduction of a new risk factor a system for stress testing automatically includes the risk factor in the user interface wherein the underlying seeded mapping structure for the new risk factor is automatically created based on seeded data entries of the risk factor.
2. The method of claim 1 wherein the risk factor includes credit rating, market rating, liquidity, operational, Probability of Default (PD), Loss Given Default (LGD), and Exposure at Default (EAD).
3. The method of claim 1 wherein the risk engine is configured for reading a base data of the at least one identified risk factor from the database.

4. The method of claim 3 wherein the engine includes Risk Weighted Asset (RWA) engine, market risk engine, financial ratio risk, Expected Credit Loss (ECL) risk, liquidity risk.
5. The method of claim 1 further comprises the step of creating an enterprise level stress scenario from a plurality of stress scenarios wherein the enterprise level stress scenario is applied to a plurality of identified risk engines of the enterprise framework for determining an enterprise level stress.
6. The method of claim 5 further comprises the step of generating a dependency data matrix of the plurality of identified risk engines wherein the dependency matrix enables triggering of the plurality of risk engines sequentially or parallelly based on a dynamic processing logic thereby reducing processing time.
7. The method of claim 1 further comprises storing a set of rules or protocols in a memory to be executed by the risk engine coupled to a processor for generating a plurality of data models associated with the stress scenario to determine the stress.
8. The method of claim 1 wherein the impact parameter is associated with the identified key risk factor based on an input received from a historical database or a real time user input or a combination of both.
9. A system for stress testing comprises:
 - an electronic user interface configured for operating on an enterprise stress testing application;
 - at least one risk factor database configured for storing a plurality of risk factors wherein a new risk factor to be associated with an enterprise framework is stored by seeding the risk factor in the database through a database initializer wherein a seed data model of the enterprise framework is configured to create an underlining seeded mapping structure for the new risk factor;
 - a controller encoded with instructions enabling the controller to function as a bot wherein the controller is configured to perform functions of:

identifying at least one key risk factor from the database based on analysis of the plurality of risk factors associated with the enterprise framework;

associating an impact parameter with the identified key risk factor;

generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter wherein a dataset join corresponding to the identified key risk factor is read by an underlying code that creates a dynamic query to fetch data needed to define the stress scenario on the risk factor; and

a processor coupled to the controller and configured for triggering at least one risk engine associated with the at least one stress scenario and the identified key risk factor for determining stress wherein the risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value;

wherein the user interface and an underlying logic of the stress testing is based on the seeded mapping wherein on introduction of the new risk factor the system automatically includes the risk factor in the user interface wherein the underlying seeded mapping structure for the new risk factor is automatically created based on seeded data entries of the risk factor.

10. The system of claim 9 wherein the risk engine is an artificial intelligence-based engine configured for processing a plurality of data models associated with the stress scenario for determining the stress.
11. The system of claim 9 wherein the risk factor includes credit rating, market rating, liquidity, operational, Probability of Default (PD), Loss Given Default (LGD), and Exposure at Default (EAD).
12. The system of claim 10 wherein the risk engine is configured for reading a base data of the at least one identified risk factor from the database.

13. The system of claim 12 wherein the engine includes Risk Weighted Asset (RWA) engine, market risk engine, financial ratio risk, Expected Credit Loss (ECL) risk, liquidity risk.
14. The system of claim 9 wherein the controller is configured for creating an enterprise level stress scenario from a plurality of stress scenarios wherein the enterprise level stress scenario is applied to a plurality of identified risk engines of the enterprise framework for determining an enterprise level stress.
15. The system of claim 9 wherein the impact parameter is associated with the identified key risk factor based on an input received from a historical database or a real time user input or a combination of both.
16. The system of claim 10 further comprises a natural language processing (NLP) server configured for processing the stress scenario based on the plurality of data models.
17. The system of claim 9 wherein the enterprise framework functions on seeded data and generates mapping structures or codes on basis of the seeded data thereby enabling inclusion of a new engine or the new risk factor within the framework upon updating of seeded entries.
18. The system of claim 14 further comprises a dependency data matrix of the plurality of risk engine configured for triggering the plurality of risk engines sequentially or parallelly depending on a dynamic processing logic.
19. The system of claim 18 wherein a plurality of components required for interacting with the plurality of identified risk engines are auto-built while creating the stress scenario, as building of these components are based on an underlying configuration of the system wherein once the stress scenario is triggered, the controller coupled to the processor interacts with the plurality of engines, determines a sequence of triggering the engines based on the dynamic processing logic, captures a stress value each from the plurality of engines, consolidates each of the stress value and provides one enterprise level stress value thereby enabling management to take appropriate action.

20. A computer-readable non-transitory storage medium storing executable program instructions for enterprise stress testing which when executed by a computer cause the computer to perform operations comprising:
- receiving by a processor at least one risk factor to be associated with an enterprise framework;
 - storing by seeding the risk factor in a risk factor database through a database initializer wherein a seed data model of the enterprise framework is configured to create an underlining seeded mapping structure for the risk factor;
 - identifying at least one key risk factor from the database based on analysis of a plurality of risk factors associated with the enterprise framework;
 - associating an impact parameter with the identified key risk factor;
 - generating at least one stress scenario based on the at least one identified key risk factor and the associated impact parameter wherein a dataset join corresponding to the identified key risk factor is read by an underlying code that creates a dynamic query to fetch data needed to define the stress scenario on the risk factor;
 - triggering at least one risk engine associated with the at least one stress scenario and the identified key risk factor for determining stress wherein the risk engine is configured for updating value of the at least one identified key risk factor based on the impact parameter to generate an impacted stress value;
 - wherein a user interface and an underlying logic of the stress testing is based on the seeded mapping wherein on introduction of a new risk factor a system for stress testing automatically includes the risk factor in the user interface wherein the underlying seeded mapping structure for the new risk factor is automatically created based on seeded data entries of the risk factor.
21. The computer-readable storage medium of claim 20 further comprises executable program instructions in a memory to be executed for generating a plurality of data models.