PSA Activities with ANDROMEDA: A Decade of Evolution, Challenges, and Training Insights

Hibti M.[1], Vermuse S.[2], Cherpin F.[2], Rolland V.[2], Picoco C.[1], Roy R.[2], Puyaubreau S.[2], Sauvaget F.[1], and Alessis B.[2]

[1]EDF R&D

[2]Engineering Units (Edvance, EDF DIPDE, EDF DT)

Abstract: ANDROMEDA Software was developed as part of the Open PSA initiative by EDF to enhance PSA activities and introduce new functionalities made possible by the standard's structure and users' expectations within EDF organization.

This paper presents a decade of feedback and experiences across various engineering business units, showcasing diverse user profiles and PSA activities. ANDROMEDA has evolved into a comprehensive software that addresses nearly all aspects of probabilistic risk assessment for complex systems.

We primarily focus on new functionalities enabled by its formalism, such as importing/exporting from/to other formalisms, qualitative sequence analysis through event sequence diagrams, event tree generation, project management and extension, version control, quality check, documentation, and scripting.

While some features have significantly boosted productivity and improved model quality (e.g., maintaining coherence between event sequence diagrams and event trees, and enhancing confidence in the development process), others remain underutilized or are perceived as too complex by some user groups (e.g., version control and embedded PSA documentation).

Certain complex functionalities have not been fully adopted in some areas, requiring users to commit to following project directives and guidelines closely. Surprisingly, the ease with which some users perform complex tasks (like using automatic generation and scripting) can inadvertently increase complexity in the PSA models unnecessarily.

To address these challenges, training sessions were organized to familiarize beginners with the tool and the wide range of functions available for their daily activities. Sessions for more experimented people were also put in place to show advanced / expert features of ANDROMEDA. This paper also reviews the evolution of these training activities, reflecting on the maturity of certain practices and the emergence of new needs among different user profiles and activities.

Keywords: Andromeda, PSA modeling and documentation, fault trees, event trees, event sequence diagrams, version control.

1. INTRODUCTION

To improve efficiency and effectivity of Probabilistic Safety Assessment (PSA) model engineering for complex systems such as nuclear power plants, a PSA tool was developed at EDF [4] to explore new modeling concepts and the state of the art of software tools around the concept of "Modular PSA" [5]. This was made possible by the introduction of the so-called "Open PSA Model Exchange Format" [6][3].

At its very beginning, ANDROMEDA aimed to introduce some missing aspects in the available commercial or internal tools that were used by the community and enhance the user experience using new functionalities (automatization, free documentation, model comparison, model verification, version control, quality control, integrated and concurrent modeling…) to help understand, develop and integrate pieces of the PSA model in a coherent and easy way.

From a research tool, ANDROMEDA was adopted indoor as one of the EDF PSA tool and was used in collaboration with RiskSpectrum for PSA activities. This paper presents a decade of feedback and experiences across various engineering business units, showcasing diverse user profiles and PSA activities.

ANDROMEDA has evolved into a comprehensive software that addresses nearly all aspects of probabilistic risk assessment for complex systems. We primarily focus on new functionalities enabled by its formalism, such as importing/exporting from/to other formalisms, qualitative sequence analysis through event sequence

diagrams, event tree generation, project management and extension, version control, quality control, documentation, and scripting.

While some features have significantly boosted productivity and improved model quality (e.g., maintaining coherence between event sequence diagrams and event trees, and enhancing confidence in the development process), others remain underutilized or are perceived as too complex by some user groups (e.g., version control) or may be a decided choice by some projects for various reasons (e.g. lack of training).

Certain complex functionalities have not been fully adopted in some areas, requiring users to commit to following project directives and guidelines closely. Surprisingly, the ease with which some users perform complex tasks (like using automatic generation and scripting) can inadvertently increase complexity in the PSA models when unnecessary.

To address these challenges, training sessions were organized to familiarize beginners with the tool and the wide range of functions available for their daily activities. Sessions for more experimented people were also put in place to show advanced / expert features of ANDROMEDA. This paper also reviews the evolution of these training activities, reflecting on the maturity of certain practices and the emergence of new needs among different user profiles and activities.

In this paper, we discuss modeling aspects, automated generation (fault trees and event trees), documentation, the scripting interface, navigation, and version control.

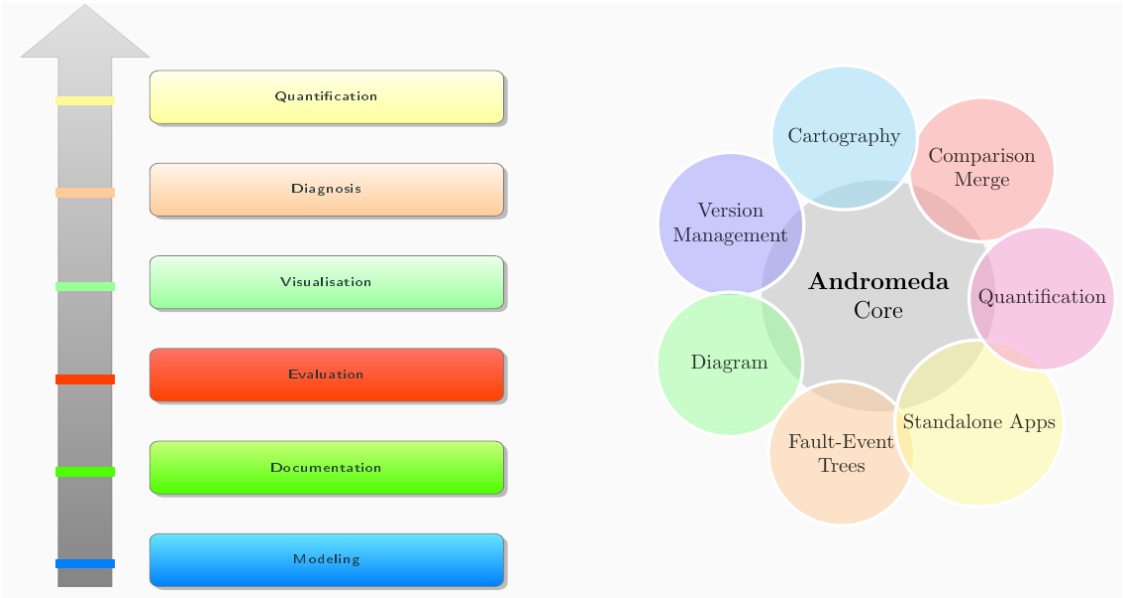

Figure 1: Andromeda Modules

2. MODELING ASPECTS

Event trees are widely used in probabilistic safety assessment for nuclear power plants. In practice, many approaches are adopted to get the event trees representing the scenarios following some initiators. This is mainly based on the work during the functional sequence analysis. This phase highlights the different outcomes of the failure and/or success of the automatic mitigation systems or human actions dedicated to reach a safe control state and/or maintain the plant in a safer state in accident/incident conditions (see [7]). In this section, we discuss the way event trees and fault trees are developed.

2.1 Event sequence diagrams and event trees

In ANDROMEDA, simple event sequence diagrams are used to describe the qualitative sequence analysis. In addition to be a good way of keeping a compact picture of the scenarios following the initiator, one can automatically generate event trees from such formalism in different forms, depending on the final researched goal: for example, the representation could be different for a Level 1 or a Level 2 PSA. We should note that

this can improve homogeneity and allow to keep the event sequence diagram (which is the real source code of the event sequences) synchronized with the associated event trees. This can also overcome the following issues:

- Event tree modeling is users dependent. Indeed, different users may lead to different event trees (considering the order of the events, their granularity and the size of the event trees).
- Some of the sequences of these event trees may need more attention to be quantified in an accurate manner with approximation algorithms.
- Sometimes the event trees present some redundant parts which means there is no optimization or that the order considered leads to useless. In practice, this can be observed by the repetition of some patterns in the event tree structure.

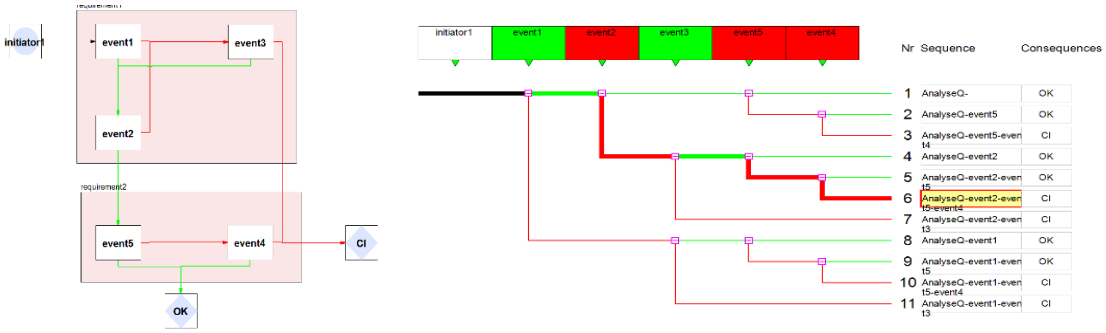

Figure 2/: An ESD with its associated event tree

Building Event Sequence Diagram (ESD) is not a trivial activity and can consume a lot of human resources. The use of an object library in ANDROMEDA is an efficient solution that allows functional analysis pieces to be made available that must be shared between several functional sequences. It is important to remember that in this ESD construction / modification activity, the need is to think at the level of safety functional requirements.

This library must be sufficiently documented to allow any user to check if the pre-coded piece of functional analysis is suitable for the functional situation they are developing, or if they should instead create a variation to take into account a different context (typically new boundary conditions, new success criteria outside the validation domain of the initial support study…).

The concept of a library can be very powerful since a library can also contain libraries of more elementary objects (initiators, functions, consequences…) facilitating shared work between users. It should be noted that the use of potentially different libraries must be documented at the very beginning of the project to share common rules (The drift of not managing libraries can lead to the creation of many very similar requirements with different names. The updating work then becomes very laborious).

The traceability of any evolution remains a major issue with its associated historization. This type of approach also allows quick checks of standardized editing rules.

One of the other interests of a dedicated ESD tooling is to quickly visualize the dependency links between shared parts of ESD. This vision allows a macroscopic vision at the level of functional requirements.

The transformation of an ESD model into an Event Trees model does not pose major difficulties. To date, the algorithm used reconstructs an exact image of the ESD modeling without seeking optimization in the cutting of the Event Trees. In the future, it may be interesting to study other transformations, such as encapsulating the content of functional requirements in the form of Fault Trees, or even more compact Event Trees constructions. These transforms may be of interest to obtain more readable Event Trees, or potentially accelerate their calculations.

The major difficulty remains entering documentation: indeed, the challenge remains to find a formalism adapted to the expected use which differs depending on the interlocutors. The ease is to produce a documentation that restores the entirety of the digital content of an ESD model. But in practice, such a rendition has little interest: it is quickly unreadable.

The work to come will consist in providing an easy formalism for the user for documenting the different pieces rather than a structured text but also defining canvases to only restore the useful information for the targeted public: for example, a communication with the people responsible for the accidental conduct requires to highlight each chronological sequence (by removing all the PSA modeling guts) by insisting on the safety objectives sought.

Another issue while drawing requirements is related to the automatic cleaning and graphic rendering tools in the sense that some deletions may have an impact beyond what is expected. A locker has been implemented to prevent such behavior.

2.2 Fault trees and KB3 generation

KB3 is an expert system based on the concept of knowledge bases that capitalizes, for a specific domain, a set of information on the system and component characteristics and behavior. This is done through several generic descriptions and rules expressed via a declarative programming language called Figaro. The Figaro language [8] is a very high-level declarative modeling language with the following objectives:

- Provide an appropriate formalism for the development of knowledge bases (with generic descriptions of components),

- Have a framework with the possibility to consider usual reliability models (e.g. traditional formalisms for static and dynamic models including Petri Nets, Markov graphs, Fault trees, Reliability block diagrams and BDMP (Boolean logic Driven Markov Processes [9]).

- Find the best trade-off between modeling power (or generality) and possibilities for the processing of models.

- Be as legible as possible and easily associated with graphic representations.

Figaro was developed combining the advantages of declarative programming with an inference engine that handles rules and the features of object-oriented programming (manipulating classes, objects, inheritance and so on).

Fault trees corresponding to failures of system missions are mainly generated from KB3.

Independently of the speed of integration faster with RiskSpectrum® PSA, the first interest of importing KB3 studies into ANDROMEDA lies in the use of ANDROMEDA's specific navigation features without any other transformation of numeric format.

The second relates to the direct use of PSA consistency tests developed to deeply verify the structural integrity of a PSA model before its subsequent quantification phase, as well as the automatic repair, the automatic completion and the slag cleaning functionalities. Help provided by consistency tests (available at each step) is a major improvement because it facilitates the detection of errors but also their localization which significantly speeds up construction of PSA models.

3. DOCUMENTATION

The construction of a PSA is a long process that takes a significant time, and its use may last for decades. At the end of the process, the full PSA model can be very complex. Therefore, a good documentation of this process as well as all the pieces of the model is of major importance for the following reasons:

- The safety systems may be subject to continuous modifications that may be demanded by the design or required by safety standards or the optimization of the operation procedures or the exploitation rules.
- There are different assumptions (for instance for the success criteria) that may change following the design/procedure modifications and the support studies.
- The previous item implies the modification updating or suppression of a number of scenarios. It follows the modification, or the entire revision of the sequence qualitative analysis diagrams.
- The data (including design flowcharts, technological classification, and material specifications) is not always available and may have some impact on the different assumptions and other considerations.
- The different reference documents are generally living with the necessity of an iterative and parallel updating.

Therefore, there is a huge amount of knowledge and references that one must consider performing his/her PSA piece. These references must be relevant and reachable in a reasonable time. These must be well documented and accessible to the PSA developers and users.

For all these reasons we can assert that a model documentation is necessary to keep possible the maintainability, transparency and quality of models that are dedicated to live dozens of years and serve as an important tool for the demonstration of safety capabilities for nuclear power plants.

In practice, most PSA models live separated of their documentation. Even if some PSA practitioners assume that there is a need of a PSA handbook to support the model, the documents focus mainly on the modeling side and may lose some relevant information related to the main assumptions and the motivations behind some tricky aspects. It is now time to propose an integrated approach with embedded documentation that has to be sufficient enough for a clear understanding of the developers or the users, but not cumbersome for a simple and easy access.

Some software tools (cf. [1] and [2]) deal with documenting PSA models. However, the approach used is based on a connection between the model and the documentation databases, which is useful for keeping documentation up to date, but they are based on database architecture which prevents from using version control in the way we intend here. We should note that our proposition may be a complementary to these two approaches.

The approach we present here is different in the sense that, we propose a way to document the model at two levels, a low level dedicated to document the modeling aspects, and a "reference" level dedicated to the documentation of the safety studies and all the necessary material for the PSA model, in addition to the analysis reports that have to be fed from the PSA records and results. Moreover, our approach is based on the concept of a modular PSA [5], which requires more documenting aspects since we may deal with many different modeling of the same aspects following the objective and application of the target PSA Model.

3.1 Free wiki documentation

There are many aspects that need to be documented in a free manner. One may want to have a logbook for some piece of the model and for this there are different ways to perform such tasks:

- Wiki related to objects: in these type of object-related wikis, one can take any notes about the object in question (it may be any object such as a parameter, a basic event, an event tree or a fault tree...) and this wiki is integrated to the object in question.
- The other notes or wikis are external notes within the PSA projects that can be seen in the folder and reached either by hyperrefs within the model or just by navigating in the folder.

Note that ANDROMEDA has an incorporated browser that allows to reach any internal or external document or link in the internet or the information system (cybersecurity aspects is handled at the corporate level by different security layers).

3.2 ESD documentation

The ESD documentation is an important point since we aim to maintain the event sequence diagram and its documentation synchronized. Indeed, in the past, the documentation which has a long cycle for being updated due to the verification and approbation process may diverge become very far from the state of the model which is quite problematic.

In ANDROMEDA, the documentation of the ESDs is now possible in the tool and reports combining the documentation and the different ESDs can be generated in the same structure that was adopted for the sequence analysis documents. Moreover, the different assumptions and references can be added to shared libraries and thus reused in other contexts (by other users for different sequences if needed) avoiding redundancies and updating incoherence problems.

Obtaining consensus on the form of the report is very complicated and automation makes lose the chronological line of accident management. This difficulty can lead to a refusal to use this module which is the heart of Andromeda.

4. ANDROMEDA BROWSER

In ANDROMEDA a web browser is integrated to help navigating within the models and their documentation. This browser connects the related pieces (for instance, an event tree and its functional events) with hyperlinks using the dedicated wikis. Recall that in ANDROMEDA, almost all the objects have their own wikis that are of two parts; the first part is generated, and includes the different links to relevant elements for the object in question, and the second part is manually edited to add information contents to the object's wiki.

The browser can also open any relevant information in the company's information system or on the web to view a document, a web site or any internal wiki file. While the browser can help, in principle, for finding information, some users prefer using the tables and are expecting the same functionalities they are accustomed to in RiskSpectrum® PSA.

While this functionality is well known it seems not fully used. The reasons may be very different but as seen in the previous section about documentation there are some difficulties to set a unified template for some documents and the users seem expecting something that fills all their needs in terms of automated report generation. But at least there are some sections that are dedicated to the bibliographic references that exist in the information system with a permanent link. Even so some users do not care about the introduction of the corresponding hyperlink.

5. MODEL COMPARISON

The main objective of model comparison is to determine the differences between models in a very exhaustive manner (no matter the source of the models). For instance, any model that is obtained through a conversion from another software after some modifications can be compared to any other version modified in ANDROMEDA.

An analyst must know if the modifications or the new pieces added to a model are really those expected and be aware about their impact on the model. When assembling different asynchronous modifications one needs to compare two versions of a model to decide how merging these pieces can be done especially in case of conflict. Sometimes the comparison can help produce a report that may automatically feed the logbook.

The major problem with any comparison lies in two points: the amount of information that has varied and the fact that some information is interdependent.

Quickly, a user can be overwhelmed by the mass of observed differences. Initially, it is important to erase the metadata related to time stamps (which are not of interest when any reading of information triggers an update of these data).

To help any user sort and gradually validate the legitimacy of the observed differences, an intermediate validation process must be used to "hide" the validated differences. But this process requires that every user masters the computer structure of the model by dealing with elementary meshes before tackling more complex meshes (ex: PARAMETER before BASIC EVENT because BASIC EVENTS can share the same PARAMETER). Therefore, help manuals must be made available to complete dedicated training in order to increase the skills of new PSA engineers.

6. VERSION CONTROL

This is one of the most complex features in ANDROMEDA. Version control allows collaboration and change tracking over time. This is valuable for teams working together, as every user can see the evolution of the model and who made which changes. It also allows reverting to previous versions if needed and recreating the exact state of a model at any point in time. This is essential for ensuring the repeatability of the results. Moreover, by comparing different versions of the model, one can identify changes that led to improvements or regressions which facilitate informed decision-making when refining the model. In ANDROMEDA, a decentralized version management was implemented which helps to prevent loss of versions of the model.

However, complex models with intricate structures or large file sizes might pose difficulty to handle merges and probably storage limitations.

The experience of using such feature was limited to a few experimented users. But even so there are many difficulties regarding the use of this functionality. The project managers are not necessarily familiar with the concept of version control.

7. API AND SCRIPTING INTERFACE

Non experimented model engineers perform most actions manually. The tools provide typically graphical user interfaces (GUI) to realize the different needed actions. But some actions have a repetitive character and when massive modifications are needed on PSA models, which requires more attention and may be very cumbersome. Introducing an application programming interface for the ANDROMEDA code allowed the possibility to interact with the models using scripting codes (e.g., python) to avoid spending time on repetitive tasks and avoid errors when performing well defined actions: the only prerequisites need to have at disposal standardized / normalized input data.

One of the main advantages of such APIs is to use programs that may be kept under version control and customized for many models instead of repeating the task many times. Recall that the comparison tools allow easily to check if the modifications were applied as expected.

Such programs can be very easily qualified if regulatory requirements asked for this level of control.

To be effective, an API must be coded in a standardized way so that the programmed functions quickly become intuitive due to a naming logic. This API must also offer advanced functions allowing to find upstream or downstream dependency links between data, this in order to more easily generate new complex data. Another interest that should not be forgotten is to allow the development of other mini-programs based on this API (without graphical interface) and aimed at covering very specific repetitive needs of some users but without burdening the native ANDROMEDA program by introducing multiple new menus and complicating its use.

The end goal of a script is to test a new requirement. Quality assurance in the nuclear world requires developing a function in ANDROMEDA but it is not desirable to keep a script for a long time.

8. TRAINING

Training has been created for the users. Two different training exist at EDF: one for the basic user and one for the advanced user.

The basic user is trained to create and use Event Sequence Diagrams , including various import and export interfacing with other tool (e.g. importing Fault Trees from KB3), model comparison and model fusion.

The advanced user is trained to use cartography, version control and scripting. This is because these three functionalities are considered to be more advanced with respect to the others. Each training lasts for two days.

Training consists in various exercises using ANDROMEDA. In particular, the version control exercise consists of a simulation with two groups of users working in parallel on a model and a model manager that has to validate the modeling (including conflicts). For using the script feature, since users are not supposed to all know how to python code, the training aims at teaching how to use the ANDROMEDA API and how to adapt existing scripts for one’s needs.

Switching to e-manual will allow for more effective training.

9. ISSUES

ANDROMEDA is a fantastic tool that integrates many rich features which significantly improve the work of PSA engineers. However, there are some areas of work to come to enhance the user experience with the tool. There are many functions and features that have been developed, sometimes at the expense of complete achievement through individual qualification of the features and of the user experience, who is sometimes a bit lost in the functionalities despite the training.

The main difficulties that have been reported and for which user workshops are ongoing are the following:

- **Version control**: It is far from being straightforward for users without background in computer science. In addition to training sessions the interface is being simplified with more graphic tools.

- **Documentation**: it suffers from the very simplicity of the wiki language. The users prefer to have synced Word documents with the wiki documentation. This choice seems suitable for the users and requires a little development.

- **ESD**: ESD design suffers from two main issues. The first is graphic, and this should be solved by a simple customization for the users depending on their screen performance. The second is about generating reports. Indeed, obtaining consensus on the form of the report is not easy depending on the projects and specific ESD objectives.

- **Navigation**: Instead of navigation through the ANDROMEDA some users prefer editing tables and are expecting the same functionalities they are accustomed to.

10. CONCLUSION AND PERSPECTIVES

In this paper, we presented several features of the ANDROMEDA Software and feedbacks related to their use at EDF in different business centers by various PSA projects. We learned many insights during this exercise. Andromeda is a complex piece of software with many functionalities in the operational of the research versions some of whom are well known to almost all the users and some are less known if not known at all. In this feedback, we focus on the main functionalities that are used or dedicated to be used for the navigation, development, the maintenance and the management of the PSA model.

While for some functionalities the users are using the tool in their daily practice without difficulties. In this feedback, we showed issues. Indeed, in addition to the classical resistance of some users (who have already their habits) there are non-intuitive functionalities that have to be explained or improved in terms of ergonomics and accessibility.

The trainings in the classical form and using different modern techniques is helpful to enhance the full use of the tool regarding the different aspects and functionalities it provides. Besides that, it turned out that these training are a good opportunity to share knowledge about the tools and their use between the users in addition to be a good place for having good requests for the next developments and needs as it is the case for the user-meetings in the PSA community (e.g. RiskSpectrum PSA and Cafta user meetings).

References

[1] PRA DocAssist, Software Manual, 2008.

[2] RiskSpectrum DOC, 2010.

[3] Epstein S, Reinhart M.F. & Rauzy A. Validation Project for the Open-PSA Model Exchange using RiskSpectrum and CAFTA. In *Proc. of the 10th International Probability assessment and Management Conference*, Seattle, 2010.

[4] Thomas Friedlhuber. *Model Engineering in a modular PSA*. PhD thesis, Ecole Polytechnique, 2014.

[5] Hibti M. Vers une {EPS} modulaire in *Actes du congrès lambda/mu*, La Rochelle, 2010.

[6] IMDR. Open {PSA} Work Package/Project P08-4 : Validation of Open-PSA Model Exchange Format for Probabilistic Safety Analyses/Project Specifications. *Technical report, Institut Pour la Maîtrise des Risques*, 2008.

[7] NRC. *Probabilistic Safety Analysis Procedure Guide*. NUREG/CR-2815, 1984.

[8] Bannelier, M., Bouissou, M., Villatte, N., & Bouhadana, H. (1991). Knowledge Modeling and Reliability Processing: Presentation of the FIGARO Language and Associated Tools. In SAFECOMP'91 (pp.). : SAFECOMP'91, Trondheim.

[9] Bouissou M. & Bon J-L. (3). A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes. Reliability Engineering and System Safety page 149-163, 82 (Issue 02), 149–163.