SECO-Sim: An Agent-Based Simulation Framework for Software Ecosystem Evolution

Francisco Victor da S. Pinheiro
Federal University of Ceará
Quixadá, Brazil
victorpinheiro@ufc.br

Emanuel F. Coutinho
Federal University of Ceará
Quixadá, Brazil
emanuel.coutinho@ufc.br

Rossana M. C. Andrade
Federal University of Ceará
Fortaleza, Brazil
rossana@ufc.br

Abstract

Software ecosystems (SECO) are commonly modeled using structural approaches such as the Software Supply Network (SSN) notation. However, existing SSN-based models are predominantly static, limiting their ability to represent ecosystem evolution and actor behavior over time. This paper presents SECO-Sim, a framework that transforms SSN models into agent-based simulation environments for SECO. The proposed approach enables the representation of dynamic events, temporal changes, and emergent behaviors through computational agents derived from SSN actors. To evaluate the feasibility of the proposal, we conducted a preliminary in-silico experiment using the SOLAR SECO under scenarios involving supplier failures, technological improvements, and platform overload. The results demonstrate that SECO-Sim preserves the original SSN structure while enabling dynamic analysis of ecosystem behavior over time. The proposal contributes toward simulation-driven SECO engineering and opens new opportunities for exploratory and predictive analyses in SECO research.

Keywords

Software Ecosystems, Simulation, Software Supply Network.

1 Introduction

A Software Ecosystem (SECO) is understood as a metaphor from Software Engineering applied to the analysis of the dynamics of software supply networks, especially those centered on platforms [7, 9, 15]. In this context, Boucharas et al. (2009) proposed the Software Supply Network (SSN) notation, widely adopted by the SECO community for modeling purposes. The SSN represents an interconnected set of software, hardware, and service organization elements that collaborate to meet market demands [6, 30]. Subsequently, this notation was expanded by Costa et al. (2013), who incorporated new components and further detailed its elements, in addition to improving the guidelines for its application.

To understand the behavior of a SECO, it is essential to analyze its organizational structure, the roles played by the actors, the relationships established between them, and the central platform that sustains the ecosystem [10, 25, 31, 32]. However, both the concept of Software Ecosystems and their applications in different domains remain in constant evolution in the contexts of research, teaching, and industry [8, 16]. In this context, the literature has pointed out several challenges and limitations related to the modeling, analysis, and use of SSN-based models for representing SECO [2, 14, 20, 24].

Despite the relevance of SSN-based modeling approaches, most existing work still has a predominantly structural and descriptive focus, concentrating on the representation of actors, relationships, and flows present in the ecosystem [19, 23, 29]. Although these representations are important for supporting the architectural and organizational understanding of SECOs, they have limitations when used for analyzing dynamic behaviors, fault propagation, operational evolution, and evaluating hypothetical scenarios over time [1, 10, 21].

Furthermore, modern SECOs have become increasingly complex, incorporating multiple vendors, distributed services, external platforms, cloud-based components, and continuous integration mechanisms [4, 26]. In this context, failures in critical actors, changes in demand, adoption of new technologies, or changes in infrastructure can generate significant impacts on the overall behavior of the ecosystem [12, 33]. However, there is still a lack of approaches capable of transforming structural SECO models into simulatable environments, allowing the execution of dynamic scenarios and the analysis of metrics related to resilience, dependency, contribution flow, structural risk, and operational capacity of the ecosystem [20, 27, 28].

Given this scenario, this work proposes SECO-Sim, an agent-based simulation framework for executing dynamic SECO scenarios modeled using the SSN notation. The proposal allows transforming structural models into simulatable environments, enabling the execution of different types of scenarios, such as critical supplier failures, increased demand, service quality degradation, and the adoption of new technologies. In addition to dynamic simulation, the framework incorporates probabilistic analysis mechanisms using Monte Carlo, allowing multiple simulations with parameter variation and evaluating the stability, sensitivity, and emergent behavior of the simulated SECO.

To evaluate the proposal, an in-silico experiment was conducted using the SOLAR SECO, in which different dynamic scenarios were executed and analyzed, including critical supplier failure, technological adoption, and platform overload. Thus, the main contributions of this work include the proposal of a dynamic simulation framework for SECO based on SSN models, the definition of mechanisms for transforming structural models into simulatable agents, the implementation of metrics for dynamic analysis of SECO, the incorporation of probabilistic simulations using the Monte Carlo method, and the execution of an in-silico experiment based on a real academic ecosystem.

The remainder of this work is organized as follows. Section 2 presents related work. Section 3 describes the SECO-Sim framework. Section 4 presents the in-silico experiment and simulation scenarios. Finally, Section 5 presents the conclusions and future work.

2 Related Work

Pinheiro et al. (2024) proposed an SSN-based approach to analyze the evolution of SECO. The work uses the *ECOS* Modeling tool to support the modeling and statistical analysis of different versions of SSN models, allowing the observation of quantitative, percentage, and average variations related to actors, relationships, and SECO components. The approach was evaluated through case studies involving the SOLAR and SIPPA SECO, as well as an expert assessment. The results indicated that the approach helps in understanding the structural evolution of ecosystems and in visualizing changes over time. Although this work advances the evolutionary analysis of SECO through SSN models, its focus remains on the structural comparison between model versions, without transforming such models into executable simulation environments. In contrast, the present study broadens this perspective by proposing SECO-Sim, capable of executing dynamic scenarios on SSN models, analyzing temporal impacts, propagating effects between actors, and conducting probabilistic experiments.

Onagh (2025) investigated the evolution of SECO and the role of community dynamics in the functional evolution of software extensions. The study analyzed 6,983 GitHub Actions from the GitHub Marketplace, identifying high functional redundancy among extensions and using graph-based analyses to understand version patterns, contributions, and evolutionary trends. Furthermore, the work evaluated the functional maturity of proprietary applications in the healthcare field, especially related to the adoption of AI-based functionalities. The results highlight limitations in functional evolution and the adoption of new capabilities in the analyzed ecosystems. Although the study proposes a data-driven approach to analyze functional evolution in SECOs, it does not address dynamic simulation mechanisms or executable modeling. In contrast, SECO-Sim proposes transforming SSN models into agent-driven simulations, allowing the exploration of evolutionary scenarios and dynamic behaviors in SECO.

Mens (2016) presented a foundational ecosystemic and socio-technical view on the maintenance and evolution of software ecosystems. The author argues that to properly address evolution challenges such as dependency management and community turnover SECOs must be analyzed as complex socio-technical networks where human contributors and technical artifacts continuously interact. Notably, the study advocates for interdisciplinary methodologies, explicitly highlighting the potential of complex systems theory and agent-based simulation to investigate the non-linear behaviors of large-scale software development. Although this work provides a strong theoretical justification for treating SECOs as complex adaptive systems and modeling developers as interacting entities, it remains a conceptual viewpoint rather than a practical tool. In contrast, SECO-Sim operationalizes this socio-technical perspective by providing a concrete framework that transforms structural SECO models into executable, agent-driven simulations, allowing for the dynamic experimentation of these interactions.

Based on the analysis of related works, it is observed that there is still a gap regarding the use of SECO structural models as a basis for simulatable environments and dynamic experimentation. Table 1 presents a comparison between the analyzed works and the proposal of this study.

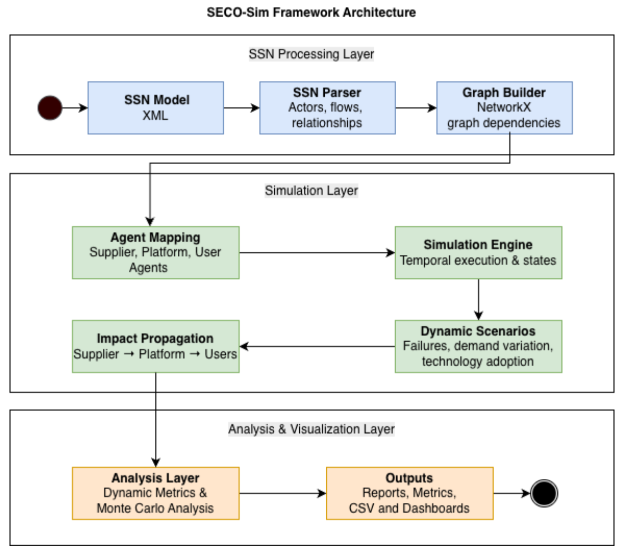

Figure 1: Overview of the SECO-Sim framework architecture for transforming SSN models into agent-based simulation environments.

3 SECO-Sim Framework

This section presents SECO-Sim, a proposed solution for transforming SECO models represented using the SSN notation into executable, agent-based simulation environments. SECO-Sim integrates mechanisms for structural transformation, temporal execution of dynamic scenarios, impact propagation, and probabilistic analysis, allowing the investigation of evolutionary behavior, dependencies, and emergent patterns in SECO.

3.1 Framework Overview

The SECO-Sim framework was designed to transform SECO structural models, represented by the SSN notation, into executable agent-based simulation environments. The proposal aims to enable the dynamic analysis of ecosystems, allowing the execution of scenarios related to impact propagation, failures, demand variations, technological adoption, and structural changes over time. Figure 1 shows the general architecture of the framework, organized into three main layers: SSN Processing Layer, Simulation Layer, and Analysis & Visualization Layer. Each layer is responsible for a specific set of activities related to the transformation, execution, and analysis of the simulatable models.

In the first layer, called SSN Processing Layer, the framework performs the initial processing of the SSN models. The process begins with the import of the model in XML format exported from the *ECOS* Modeling tool, containing actors, flows, relationships, and dependencies of the ecosystem. Next, the SSN Parser module interprets the structural elements of the model, extracting information relevant to the simulation. Subsequently, the Graph Builder module converts the model into a graph-based representation, using structures compatible with network analysis libraries, allowing the representation of dependencies, connections, and flows between the actors in the SECO.

Table 1: Comparison between related work and the proposed study

Work	Objective	Approach	Focus	Difference Compared to the Proposed Study
Pinheiro et al. (2024)	Analyze the evolution of SECO using SSN models and statistical comparisons between ecosystem versions	Structural modeling + statistical analysis	Evolution of SSN models	Focuses on structural comparison between model versions, without executable simulation or dynamic scenario execution
Onagh (2025)	Investigate the influence of community dynamics on the functional evolution of software extensions	Graph analysis + empirical ecosystem analysis	Functional evolution in SECO	Focuses on functional redundancy and community evolution, without dynamic simulation or executable SECO models
Mens (2016)	Present a socio-technical view on the challenges of software ecosystem maintenance and evolution	Conceptual analysis + interdisciplinary review	Socio-technical dynamics in SECO	Provides a theoretical foundation and suggests agent-based simulation, but lacks a concrete, executable simulation framework
Proposed Study	Propose a simulation framework for SECO based on SSN models	Agent-based simulation + Monte Carlo + SSN transformation	Dynamic simulation of SECO	Transforms SSN models into executable simulation environments, enabling dynamic scenario execution, probabilistic analysis and behavioral evaluation of ecosystems over time

The second layer, called Simulation Layer, contains the mechanisms responsible for running the simulation. Initially, the Agent Mapping module transforms the elements of the SSN model into simulatable agents, representing actors such as suppliers, platforms, users, and ecosystem components. These agents then possess specific states, attributes, and behaviors during the simulation execution.

The Simulation Engine module is responsible for the temporal control of execution, managing states, events, and interactions between agents over time. Through this mechanism, the framework enables the execution of different dynamic scenarios, including failures in critical actors, variations in demand, adoption of new technologies, and structural changes in the SECO. Furthermore, the Impact Propagation module allows for the analysis of how events and changes in specific actors can propagate through the ecosystem network, affecting other connected components. This mechanism makes it possible to investigate critical dependencies, structural resilience, and potential cascading effects within the SECO.

Finally, the Analysis & Visualization Layer is responsible for the analytical processing of the results generated during the simulation. The Analysis Layer module performs the calculation of dynamic metrics and probabilistic analyses using Monte Carlo simulations, allowing the observation of emerging patterns, SECO stability, and agent behavior under different experimental conditions. The results generated by the framework can be exported in different formats, including reports, statistical metrics, CSV files, and analytical dashboards, offering support for experimental analysis and understanding the dynamic behavior of SECO.

3.2 SSN-to-Agent Transformation

The SSN-to-Agent transformation stage constitutes the structural core of SECO-Sim. In this stage, the elements represented in the SSN models are converted into executable entities capable of participating in the simulation environment. The objective of this transformation is to allow originally static components of the modeling to acquire dynamic behavior, internal states, and the ability to interact over time. Initially, the SSN model is processed by the framework's parser, responsible for identifying actors, relationships, dependencies, flows, and structural components of the SECO. Then, this information is used to construct a graph representation of the ecosystem, allowing the mapping of connections and dependency relationships between the different elements.

After the graph is constructed, the framework performs the mapping process to agents. Each actor in the SECO can be transformed into a computational agent with specific characteristics. For example, suppliers can represent agents responsible for providing services or components; platforms can act as central integration elements; users and consumers can represent demand agents; while intermediaries can perform propagation or coordination functions.

In addition to agent types, the transformation process also defines behavioral and structural attributes, such as dependency level, criticality, contribution capacity, degree of connectivity, and operational state. These attributes are subsequently used during simulation execution to define interaction patterns, impact propagation, and temporal evolution of the SECO. The SSN-to-Agent transformation allows preserving the original structure of the SSN model while adding the dynamic capabilities necessary for running simulatable scenarios. In this way, the framework makes it possible to investigate not only the structural composition of SECO, but also the emergent behavior resulting from the interactions between its actors. Table 2 presents the main agent types currently supported by SECO-Sim, as well as their respective structural mappings and simulatable behaviors.

Table 2: Agent types supported by SECO-Sim

Agent Type	SSN Mapping	Behavior in Simulation
Supplier Agent	Suppliers / Organizations	Provides services and components
Platform Agent	Platforms / SECO Core	Coordinates interactions and dependencies
Consumer Agent	Clients / Users	Generates demand and consumes services
Intermediate Agent	Aggregators / Intermediaries	Propagates flows and mediates interactions

3.3 Simulation Process

The SECO-Sim simulation process is responsible for the dynamic execution of the modeled SECOs, allowing for the analysis of the temporal evolution of agents, the propagation of events, and the impacts generated by different scenarios. The simulation occurs in discrete temporal cycles, in which agents interact according to rules defined by the environment and configured parameters. During each execution cycle, the framework updates the state of the agents, processes active events, recalculates dependencies, and records metrics related to the SECOs behavior. This process makes it possible to observe phenomena such as fault propagation, platform overload, changes in contribution flows, demand changes, and effects resulting from the adoption of new technologies. The simulation process is divided into four main stages: definition of input models, scenario configuration, event propagation, and temporal execution flow. These stages allow for structuring the simulation execution in a modular and extensible way.

3.3.1 Input Models. SECO-Sim input models are based on SSN representations exported in structured formats, such as XML or JSON. These models describe the SECO actors, their relationships, dependencies, and interaction flows. The elements extracted from the models are used as the basis for building the simulation environment. From this information, the framework identifies the types of agents, the existing links between them, and the possible paths for impact propagation in the SECO. In addition to structural information, the input models can include additional metadata, such as criticality levels, dependency weights, interaction frequency, and actor relevance indicators. This data allows for enriching the simulation behavior and making the scenarios more closely resemble real-world environments.

3.3.2 Scenario Configuration. Scenario configuration allows you to define the experimental conditions that will be executed during the simulation. This step makes it possible to create different situations capable of altering the behavior of the SECO over time. Among the scenarios supported by the framework are failures in critical components, service interruptions, increases or decreases in demand, the insertion of new actors, the adoption of emerging technologies, and changes in the levels of dependence between agents. Each scenario can have specific parameters, such as event duration, impact intensity, probability of occurrence, and scope of affected agents. This flexibility allows for the execution of different types of experiments and the evaluation of ecosystem behavior under varying conditions.

3.3.3 Event Propagation. The event propagation mechanism is responsible for disseminating impacts among SECO agents during simulation execution. When an event occurs in a particular agent, its effects can propagate to other agents connected by dependency or flow relationships. Propagation considers factors such as the criticality of the relationships, the intensity of the impact, the degree of connectivity, and the current state of the affected agents. In this way, local events can generate systemic effects in different parts of the ecosystem. This mechanism allows for the analysis of phenomena such as fault cascades, disruption of contribution flows, degradation of central platforms, and effects resulting from excessive dependence on critical actors. Furthermore, the propagation process makes it possible to investigate levels of resilience and recovery capacity of the ecosystem.

3.3.4 Execution Flow and Monte Carlo Support. The simulation execution flow is organized into discrete temporal cycles. In each cycle, the framework executes a set of steps responsible for updating the ecosystem's state. Initially, active events are processed and propagated among connected agents. Then, the internal state of each agent is updated according to its interactions, dependencies, and operational parameters. After this update, dynamic metrics are recalculated and recorded for later analysis. The flow continues until the simulation's stopping condition is reached, such as the maximum number of cycles, ecosystem stabilization, or the end of the configured events. At the end of the execution, the results are stored in structured formats, enabling the generation of reports, graphs, and statistical analyses.

SECO-Sim incorporates support for probabilistic simulations using the Monte Carlo method [13], allowing multiple iterations of the same scenario with controlled random variations. The goal is to reduce dependence on deterministic results and enable more robust analyses of SECO behavior. During Monte Carlo runs, parameters such as failure probability, impact intensity, agent behavior, and event propagation can vary according to predefined probability distributions. Each run produces distinct results, allowing the observation of recurring patterns, average behavior, and extreme scenarios. The use of Monte Carlo makes it possible to assess the resilience, stability, and sensitivity levels of the SECO in the face of operational uncertainties. Furthermore, the framework can calculate aggregate statistical indicators, such as mean, standard deviation, and frequency of occurrence of certain behaviors throughout the simulations.

3.4 Dynamic Metrics

SECO-Sim provides a set of dynamic metrics designed for the behavioral and evolutionary analysis of SECO during simulation execution. These metrics allow observation of how agents and relationships evolve over time in different scenarios. Supported metrics include impact propagation, dynamic centrality, failure rate, component availability, structural stability, interaction density, contribution volume, and mean ecosystem recovery time.

Metrics are calculated continuously during execution cycles and stored for later statistical analysis and visualization. This allows for the identification of emerging patterns, structural bottlenecks, critical agents, and ecosystem resilience levels. In addition to individual metrics, the framework also allows for combining indicators to generate composite analyses of systemic behavior, facilitating an understanding of the overall ecosystem dynamics and supporting experimental studies related to evolution, sustainability, and structural dependency in SECO. Table 3 presents some of the main metrics currently supported.

Table 3: Dynamic metrics supported by SECO-Sim

Metric	Description
Impact Propagation Rate	Measures how failures spread through the SECO
Recovery Time	Average time required for SECO stabilization
Dynamic Centrality	Measures the temporal importance of agents
Failure Rate	Percentage of agents affected during execution
Interaction Density	Volume of active interactions between agents

4 Preliminary In-Silico Experiment

As a preliminary assessment of the SECO-Sim framework's functionality, an in-silico experiment was conducted using the SOLAR SECO.

4.1 SECO Description

As a preliminary study for the evaluation of SECO-Sim [11], SOLAR SECO, a Virtual Learning Environment developed by Federal University of Ceara, was used, focusing on supporting face-to-face, blended, and distance learning courses. SOLAR was conceived as an educational platform based on open-source software and oriented towards integration with different systems, services, and technologies, constituting over time a complex and multifaceted SECO. SOLAR SECO emerged from the continuous evolution of the main platform and the formation of a network of actors responsible for providing technologies, services, integrations, maintenance, content, and complementary functionalities [9, 22].

The SECO system has a strong dependence on external technological components, including web servers, databases, development frameworks, and institutional academic systems. Among the main technological components used are PostgreSQL, NGINX, Linux,

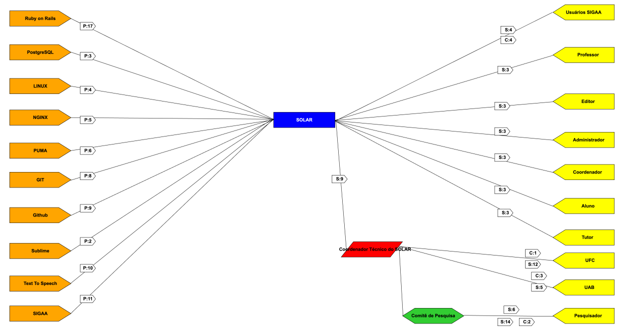

Figure 2: SSN model of the SOLAR SECO [8].

Puma, GitHub, and Ruby on Rails, in addition to integration with academic systems such as SIGAA. SOLAR also provides APIs that allow integration with external applications, contributing to the growth of a community of developers and services associated with the platform. Figure 2 presents the SOLAR SECO SSN model, highlighting the main actors, relationships, and flows of the SECO.

The choice of the SOLAR SECO for the in-silico experiment was due to its structural complexity, diversity of actors, and the presence of multiple technological and organizational dependencies. Furthermore, this SECO had already been previously modeled using the SSN notation, which facilitated its transformation into an agent-based model within the SECO-Sim context. The SSN model used in this experiment contains 49 elements, including 23 actors, 26 interaction flows, and 22 business relationships. Among the actors are SOLAR as the company of interest, 10 suppliers, 10 clients/users, one intermediary, and one aggregator. All actors were processed by the SECO-Sim parser and converted into 23 computational agents, enabling the execution of dynamic simulation scenarios involving failures, overloads, and structural changes in the SECO.

4.2 Simulation Scenarios

To evaluate the behavior of the modeled SOLAR SECO under different conditions, three simulation scenarios were defined: Critical Supplier Failure, Technology Adoption Improvement, and Platform Overload. Each scenario introduces specific events over time, allowing observation of structural and behavioral changes in SECO. The Critical Supplier Failure scenario simulates failures in important technology suppliers, such as PostgreSQL and NGINX, as well as reductions in platform quality and user satisfaction. The objective is to analyze the SECOs resilience in the face of the unavailability of critical dependencies. The Technology Adoption Improvement scenario represents a positive evolution of the ecosystem through improvements in technology quality and increased user demand. This scenario seeks to evaluate how technological strengthening impacts the contribution flow, adoption rate, and SECO growth. Finally, the Platform Overload scenario simulates accelerated growth in user demand combined with degradation of platform quality and reduced user satisfaction.

4.3 Simulation Results

The simulations were run for 30 time cycles for each scenario, using the same SSN model of the SOLAR SECO. In all scenarios, the model was converted into 23 simulation agents, including 10 vendors, 8 users, 2 clients, 1 platform, 1 intermediary, and 1 aggregator.

In the Critical Supplier Failure scenario, the results indicate a progressive reduction in the SECOs operational capacity after the deactivation of critical suppliers. The PostgreSQL failure in step 10 reduced the dependency level from 1.0 to 0.8693 and increased the dependency risk to 0.2. Subsequently, the NGINX failure in step 18 again reduced the dependency level to 0.7582 and increased the risk to 0.4. At the end of the simulation, the contribution flow was 45.3279, the adoption rate was 0.4886, and the final backlog was 80.1037. These results indicate that failures in structural suppliers directly affect the overall performance of the SECO and reduce its ability to maintain stable flows.

In the Technology Adoption Improvement scenario, a more positive behavior was observed. Quality improvements in actors such as GitHub, PostgreSQL, NGINX, SOLAR, Tutor, and SIGAA Users contributed to maintaining the dependency level at 1.0 and the dependency risk at 0.0 throughout the execution. The final contribution stream reached 87.5467, while the final adoption rate was 0.593. However, the increased demand also raised the platform load to 19.3602 and the final backlog to 183.6021, indicating that technological improvements can expand ecosystem activity but also generate greater operational pressure on the core platform.

In the Platform Overload scenario, the initial surge in demand temporarily increased the contribution flow, which reached 97.1792 in step 7. However, the gradual degradation of platform and supplier quality, combined with reduced user satisfaction, caused a continuous decline in the indicators. At the end of the simulation, the contribution flow fell to 36.7984, the adoption rate to 0.3776, and the platform load to 8.1024. These results suggest that overload, when combined with loss of quality and satisfaction, can compromise the sustainability of the SECO over time.

A comparative analysis of the scenarios reveals different SECO behaviors under distinct operational conditions. In the critical supplier failure scenario, the deactivation of critical suppliers reduced the dependency level from 1.0 to 0.7582 and increased the dependency risk to 0.4, directly impacting the SECOs stability. As a consequence, the final contribution flow fell to 45.3279 and the adoption rate decreased to 0.4886. Figure 3 shows a comparison of the evolution of the contribution flow between the simulated scenarios.

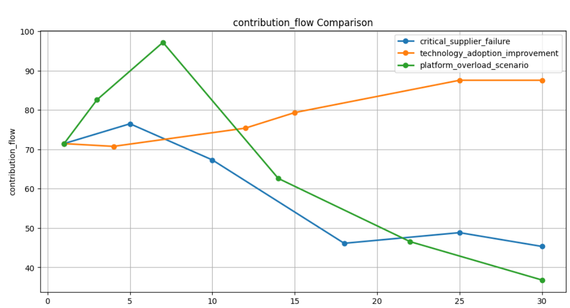

Figure 3: Comparison of contribution flow evolution across the simulated scenarios.

4.4 Discussion

The results obtained in the in-silico experiment demonstrate the potential of SECO-Sim to support dynamic analyses in SECO using SSN models. Unlike purely structural approaches, the framework allows for the representation of temporal changes, failures, overloads, and alterations in actor behavior over time, making it possible to observe emerging impacts on the SECO. The simulated scenarios showed that critical dependencies can significantly compromise the stability of the ecosystem, while technological improvements tend to strengthen contribution flows and platform adoption, although they can also increase operational pressure on the core infrastructure. Furthermore, the experiment showed that the framework is able to preserve the structure of the original SSN model during the transformation to computational agents, allowing for the connection of structural modeling and dynamic simulation in the same process. These results indicate that SECO-Sim can be used to support exploratory analysis, resilience assessment, and the investigation of complex behaviors in software ecosystems.

5 Conclusion and Future Work

This work presented SECO-Sim, a framework for transforming SSN models into agent-based simulation environments for SECOs. The framework extends static SSN modeling by enabling the simulation of dynamic events, temporal changes, and emergent behaviors. As a preliminary evaluation, SECO-Sim was applied in an in-silico experiment using the SOLAR SECO under scenarios involving supplier failures, technological improvements, and platform overload. The results showed that the framework preserved the original SSN structure while successfully representing dynamic SECO behavior over time. From a practical perspective, these simulated metrics can assist SECO managers, organizations, and communities in strategic decision-making, such as identifying critical vulnerabilities, anticipating bottlenecks caused by rapid technology adoption, and planning resource allocation proactively.

This study has some threats to validity. The evaluation was conducted using a single preliminary in-silico experiment, which limits the generalizability of the results to other SECOs. Furthermore, the simulation parameters—such as failure intensity, degradation thresholds, and impact weights—were defined manually without empirical justification from real-world operational data. To address these limitations, future work will focus on calibrating these parameters using historical data from Mining Software Repositories (MSR) and expert elicitation. We also plan to fully exploit the framework's Monte Carlo capabilities through aggregated probabilistic runs, expand the agent models to support gateways, and evaluate SECO-Sim across diverse, concrete target SECOs with well-defined evaluation criteria.

Artifact Availability

The data collected and used in this work are publicly available at: https://doi.org/10.5281/zenodo.20298342. The repository also includes additional graphs, reports, analyses, and Python scripts used in the experiments.

References

[1] Alencar, Igor R. and Coutinho, Emanuel F. and Moreira, Leonardo O. and Bezerra, Carla I. M. 2020. A Tool for Software Ecosystem Models: An Analysis on Their Implications in Education. In *Proceedings of the 34th Brazilian Symposium on Software Engineering* (Natal, Brazil) *(SBES '20)*. Association for Computing Machinery, New York, NY, USA, 405–414.

[2] L. M. Ambrósio, P. Marques, J. M. N. David, R. Braga, M. A. Ribeiro Dantas, V. Ströele, and F. Campos. 2019. An Approach to Support Data Integration in a Scientific Software Ecosystem Platform. In *2019 IEEE 23rd International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. 39–44.

[3] Vasilis Boucharas, Slinger Jansen, and Sjaak Brinkkemper. 2009. Formalizing Software Ecosystem Modeling *(IWOCE '09)*.

[4] Thommie Burström, Tom Lahti, Vinit Parida, Markus Wartiovaara, and Joakim Wincent. 2024. Software Ecosystems Now and in the Future: A Definition, Systematic Literature Review, and Integration Into the Business and Digital Ecosystem Literature. *IEEE Transactions on Engineering Management* 71 (2024), 12243–12258. https://doi.org/10.1109/TEM.2022.3216633

[5] Gabriella Costa, Felyppe Silva, Rodrigo Santos, Cláudia Werner, and Toacy Oliveira. 2013. From Applications to a Software Ecosystem Platform: An Exploratory Study. In *Proceedings of the Fifth International Conference on Management of Emergent Digital EcoSystems*.

[6] Emanuel Ferreira Coutinho and Carla Ilane Moreira Bezerra. 2019. Um Estudo sobre a Variabilidade de Aspectos Dinâmicos no Ecossistema de Software Educacional SOLAR. In *Anais do I Workshop em Modelagem e Simulação de Sistemas Intensivos em Software* (Salvador). SBC, Porto Alegre, RS, Brasil, 59–68. https://doi.org/10.5753/mssis.2019.7560

[7] Emanuel F. Coutinho, Italo Santos, Leonardo O. Moreira, and Carla I. M. Bezerra. 2019. A Report on the Teaching of Software Ecosystems in Software Engineering Discipline. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering* (Salvador, Brazil) *(SBES '19)*. Association for Computing Machinery, New York, NY, USA, 130–139. https://doi.org/10.1145/3350768.3351302

[8] Emanuel Ferreira Coutinho, Italo Santos, and Carla Ilane Moreira Bezerra. 2017. A Software Ecosystem for a Virtual Learning Environment: SOLAR SECO. In *2017 IEEE/ACM Joint 5th International Workshop on Software Engineering for Systems-of-Systems and 11th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (JSOS)*. 41–47. https://doi.org/10.1109/JSOS.2017.2

[9] Emanuel Ferreira Coutinho, Davi Viana, and Rodrigo Pereira Dos Santos. 2017. An Exploratory Study on the Need for Modeling Software Ecosystems: The Case of SOLAR SECO. In *2017 IEEE/ACM 9th International Workshop on Modelling in Software Engineering (MiSE)*. 47–53. https://doi.org/10.1109/MiSE.2017.3

[10] Coutinho, Emanuel e Bezerra, Carla. 2020. A study on dynamic aspects variability in the SOLAR educational software ecosystem. *Journal of the Brazilian Computer Society* 26, 1 (2020), 1–19.

[11] Breno Bernard Nicolau de França and Guilherme Horta Travassos. 2016. Experimentation with dynamic simulation models in software engineering: planning and reporting guidelines. *Empirical Software Engineering* 21, 3 (2016), 1302–1345.

[12] Shady Hegazy, Christoph Elsnere, Jan Bosch, and Helena Holmström Olsson. 2024. Experimentation in Industrial Software Ecosystems: an Interview Study. In *2024 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. 97–105. https://doi.org/10.1109/SEAA64295.2024.00023

[13] M Hénon. 1971. The Monte Carlo method. In *International Astronomical Union Colloquium*, Vol. 10. Cambridge University Press, 151–167.

[14] Slinger Jansen. 2020. A focus area maturity model for software ecosystem governance. *Information and Software Technology* 118 (2020), 106219. https://doi.org/10.1016/j.infsof.2019.106219

[15] Slinger Jansen, Eko Handoyo, and Carina Alves. 2015. Scientists' needs in modelling software ecosystems. In *Proceedings of the 2015 European Conference on Software Architecture Workshops*. 1–6.

[16] John D McGregor. 2010. A method for analyzing software product line ecosystems. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*. 73–80.

[17] Tom Mens. 2016. An Ecosystemic and Socio-Technical View on Software Maintenance and Evolution. In *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. 1–8. https://doi.org/10.1109/ICSME.2016.19

[18] Elmira Onagh. 2025. Evolving Software Ecosystems: The Role of Community Dynamics in Shaping Software Extensions. (2025). https://doi.org/10315/43077

[19] Francisco Pinheiro, Emanuel Coutinho, Ronier Lima, Maria Silva, Carla Bezerra, and Rossana Andrade. 2025. ECOS Modeling: A Modeling Tool, Repository for Models and Evolution Analysis of Software Ecosystems. In *Anais do XXI Simpósio Brasileiro de Sistemas de Informação* (Recife/PE). SBC, Porto Alegre, RS, Brasil, 369–378. https://doi.org/10.5753/sbsi.2025.246501

[20] Francisco Pinheiro, Emanuel Coutinho, Marcelo Silva, Sidartha Carvalho, and Rossana Andrade. 2025. Investigando a Integração Estrutural de LLMs em Ecossistemas de Software: Um Estudo com Modelagem SSN. In *Anais do XXXIX Simpósio Brasileiro de Engenharia de Software* (Recife/PE). SBC, Porto Alegre, RS, Brasil, 832–838. https://doi.org/10.5753/sbes.2025.11601

[21] Francisco Pinheiro, Maria Silva, Ronier Lima, Emanuel Coutinho, Carla Bezerra, and Rossana Andrade. 2025. Estudo Preliminar sobre a Modelagem de Ecossistemas de Software: O Caso do ECOS SIPPA. In *Anais do XX Simpósio Brasileiro de Sistemas Colaborativos* (Manaus/AM). SBC, Porto Alegre, RS, Brasil, 15–27. https://doi.org/10.5753/sbsc.2025.6553

[22] Francisco Victor Pinheiro and Emanuel Coutinho. 2024. Uma Abordagem Baseada em Rede de Fornecimento de Software para Análise da Evolução de Ecossistemas de Software. In *Anais Estendidos do XX Simpósio Brasileiro de Sistemas de Informação* (Juiz de Fora/MG). SBC, Porto Alegre, RS, Brasil, 138–153. https://doi.org/10.5753/sbsi_estendido.2024.238718

[23] Francisco Victor Pinheiro, Emanuel Coutinho, Maria Erilane Silva, and Carla Bezerra. 2024. A Systematic Mapping of Health, Quality, Evolution, Simulation and Modeling in Software Ecosystems. In *Proceedings of the 20th Brazilian Symposium on Information Systems* (Juiz de Fora, Brazil) *(SBSI '24)*. Association for Computing Machinery, New York, NY, USA, Article 26, 10 pages. https://doi.org/10.1145/3658271.3658297

[24] Francisco Victor da S. Pinheiro, Emanuel Ferreira Coutinho, Italo Santos, and Carla I. M. Bezerra. 2022. A Tool for Supporting the Teaching and Modeling of Software Ecosystems Using SSN Notation. *Journal on Interactive Systems* 13, 1 (Sep. 2022), 192–204. https://doi.org/10.5753/jis.2022.2602

[25] Rodrigo Santos, George Valença, Davi Viana, Bernardo Estácio, Awdren Fontão, Sabrina Marczak, Cláudia Werner, Carina Alves, Tayana Conte, and Rafael Prikladnicki. 2014. Qualidade em ecossistemas de software: Desafios e oportunidades de pesquisa. In *Proceedings of VIII Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems*. 41–44.

[26] André Silva, Francisco Pinheiro, and Emanuel Coutinho. 2025. Uma Abordagem Baseada em Modelos SSN para Análise da Saúde e Qualidade de Ecossistemas de Software. In *Anais Estendidos do XXI Simpósio Brasileiro de Sistemas de Informação* (Recife/PE). SBC, Porto Alegre, RS, Brasil, 379–385. https://doi.org/10.5753/sbsi_estendido.2025.246849

[27] André Luís Silva, Francisco Victor Pinheiro, Emanuel Coutinho, Rhenara Alves, and Rossana Andrade. 2026. Análise Estrutural da Saúde de Ecossistemas de Software e suas Implicações para Sistemas Colaborativos: Um Estudo de Caso Baseado em SSN do ECOS SOLAR. In *Anais do XXI Simpósio Brasileiro de Sistemas Colaborativos* (Porto Alegre/RS). SBC, Porto Alegre, RS, Brasil, 1–12. https://doi.org/10.5753/sbsc.2026.19698

[28] André Luís Silva, Francisco Victor Pinheiro, Emanuel Coutinho, and Rossana Andrade. 2026. Avaliação da Saúde e Qualidade do Ecossistema de Software SIPPA Utilizando Modelagem SSN. In *Anais Estendidos do XXII Simpósio Brasileiro de Sistemas de Informação* (Vitória/ES). SBC, Porto Alegre, RS, Brasil, 250–260. https://doi.org/10.5753/sbsi_estendido.2026.249061

[29] Maria E. Silva, Francisco Pinheiro, Carla Bezerra, and Emanuel Coutinho. 2023. Modelagem de Ecossistemas de Software das Plataformas de Computação em Nuvem AWS e GCP. In *Anais Estendidos do XIX Simpósio Brasileiro de Sistemas de Informação* (Maceió/AL). SBC, Porto Alegre, RS, Brasil, 172–177. https://doi.org/10.5753/sbsi_estendido.2023.229306

[30] P. R. Silva. 2018. Um enfoque da identificação de papéis e modelagem de ecossistemas de software em um sistema de gestão acadêmica. Monografia (Bacharel em Ciência da Computação), UFRR (Universidade Federal de Roraima), Boa Vista, Brazil.

[31] Ivo van den Berk, Slinger Jansen, and Lútzen Luinenburg. 2010. Software Ecosystems: A Software Ecosystem Strategy Assessment Model. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume* (Copenhagen, Denmark) *(ECSA '10)*. Association for Computing Machinery, New York, NY, USA, 127–134. https://doi.org/10.1145/1842752.1842781

[32] Jelle Wouters, JR Ritmeester, AW Carlsen, Slinger Jansen, and Krzysztof Wnuk. 2019. A SECO meta-model. In *International Conference on Software Business*. Springer, 31–45.

[33] Rodrigo Oliveira Zacarias, Léo Carvalho Ramos Antunes, Márcio de Oliveira Barros, Rodrigo Pereira dos Santos, and Patricia Lago. 2025. Exploring developer experience factors in software ecosystems. *Journal of Systems and Software* 230 (2025), 112549. https://doi.org/10.1016/j.jss.2025.112549