UDC 004.413 https://doi.org/10/15407/pp2022.03-04.092

TOWARD SOFTWARE ENGINEERING ECOSYSTEMS DEFINITION

Mykola Sydorov

Nowadays, the fundamental science of software engineering is being formed. It should represent knowledge that meets the requirements of the concept of sustainable development. This fundamental science could be named as the Software Engineering Ecology. Along with others, it should include a section containing knowledge about software engineering ecosystems. This section of the future science has been intensively developing for more than fifteen years. However, today, there is no consensus among researchers regarding the definitions of the software ecosystem. Naturally, it does not contribute to the creation of an appropriate section of emerging science. All authors offer only a software ecosystem, considering it in different contexts and defining it in different ways. The term "software ecosystem" is now used to refer to a wide range of ecosystems that are actually software engineering ecosystems. The purpose of this paper is to propose a basis for defining software engineering ecosystems. By analogy with the concepts of the landscape and the trophic chain of biological ecosystems, the concepts of software landscape and software engineering value chain are proposed. Based on these concepts, the diversity of software engineering ecosystems is shown. A model of the software engineering ecosystems and a classification of the software engineering ecosystems are proposed.

Keywords: Software ecosystem, software engineering, landscape, value chain, ecosystem model, ecosystems types

Introduction

Software as the result of software engineering is always a product. It has a user and an operating environment. The product is created and transferred to the customer or buyer in the context of the life cycle and must meet a number of requirements specific to products of any engineering (product design, quality, standards, documentation, economics, maintenance, environmental impact in the context of sustainable development). The life cycle is a system-forming factor in software engineering, as it defines the processes, resources and products used and created by engineering. This feature of software engineering also determines the structure of knowledge inherent in it, which can be represented as a layered cylinder. The layers of the cylinder represent the fundamental sciences of software engineering, and the vertical division corresponds to the knowledge regarding the practical implementation of the phases of the software life cycle. It is obvious that the fundamental sciences are applied in any vertical section of the cylinder. For example, the fundamental sciences include Software engineering economics, Software engineering culture, Computing Foundations [1]. The fundamental sciences should also include the now emerging science, which, by being included in the vertical sciences, should supply knowledge that meets the requirements of the concept of sustainable development. This may be the Software Engineering Ecology [2, 3]. Along with others, the Software Engineering Ecology should include a section containing knowledge about software engineering ecosystems. This section of the future science has been intensively developing for more than fifteen years. However, today, there is no consensus among researchers regarding the definitions of the software ecosystem. Naturally, this does not contribute to the creation of an appropriate section, an emerging science. All authors offer only a software ecosystem, considering it in different contexts and defining it in different ways. Based on the hypothesis that the term "software ecosystem" is now used to refer to a wide range of ecosystems that are actually software engineering ecosystems, the purpose of this paper was to proposed a basis for defining software engineering ecosystems. As such a base, by analogy with the concepts of the landscape and the trophic chain of biological ecosystems, the concepts of software landscape and software engineering value chain are proposed. Based on these concepts, the diversity of software engineering ecosystems is shown. A model of the software engineering ecosystems and a classification of the software engineering ecosystems are proposed.

The software engineering ecosystems diversity

The software engineering ecosystem landscape. According to the definition of ecology, an ecosystem (biogeocenosis) is defined as a supraorganismal system of interacting biotic (living) and abiotic (non-living) components located in a certain territory [4, 5]. In ecology, an ecosystem is always defined, a certain area, space, terrain, landscape. For example, the ecosystem of a forest, lake, pond. It is an important component of the ecosystem. Therefore, when defining the software engineering ecosystem, it is expedient to define a similar concept.

Four candidates can be used as a metaphor for the similar concept of the software engineering ecosystem. These are territory, environment, terrain and landscape. The term "territory" usually refers to a piece of space that can be clearly defined. Usually, this is the territory of the country. This term is proposed to be applied at the level of Software engineering. The term "environment" has long and consistently been used for the software development environment [6]. The term "terrain" refers to a specific space that is already landscaped with the biotic com-

ponent of the ecosystem. In addition, it is used in geographic information systems when visualizing space. This term can be analogous to biotope [5]. Finally, the term "landscape" is used to refer to a space, whether or not it is landscaped. In the software engineering, this term is used as a metaphor for the visualization of a software system [7]. In our opinion, it is this term that is most suitable for designating the software engineering ecosystem space, taking into account its initial independence from the activity of the biotic component. Thus, initially, the landscape, being an ecotope, as a result of transformations (activities) carried out by the biota is transformed into a biotope, turning, for example, into a terrain. In this context, this transformation is important for two reasons. First, unlike ecology, the landscape of an ecosystem in software engineering cannot initially be specified (it is not a forest, a field, a pond, etc.). However, this can be done if the landscape connect to the activity of the biota. Secondly, the activity of the biota will determine the nature of the biotope in the future. Therefore, biotopes will be different for different ecosystems even with the same activity.

The software engineering value chain. The ecosystem of biology is defined as the unity of interacting biotic (living) and abiotic (non-living) components, which, based on the energy flow, establishes the trophic structure of the ecosystem, species diversity and the circulation of substances in it [4]. Therefore, the second important part of an ecosystem is its trophic structure. In defining the software engineering ecosystem, we will follow this definition. However, instead of the trophic structure, we will use the added value factor of the same importance for software engineering. Then, instead of the trophic structure in software engineering, we will use the value chain [8].

To build a chain, we will use nodes of two types (Fig. 1.). The first type of node designates the activity that corresponds to the process of the chain (Fig. 1, a), indicating in it the designation of the process (Activity) and indirectly the landscape, the designation of the biotic component (Biota), which implements the process and the abiotic component (Abiota), which the biotic component uses to implement the process. The second type of node, we use to denote the added value - AVi (Fig. 1, b).

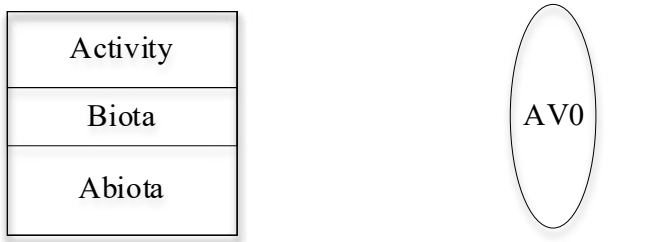

Fig. 1. Types of value chain nodes

The value chain for software engineering can look like this (Fig. 2).

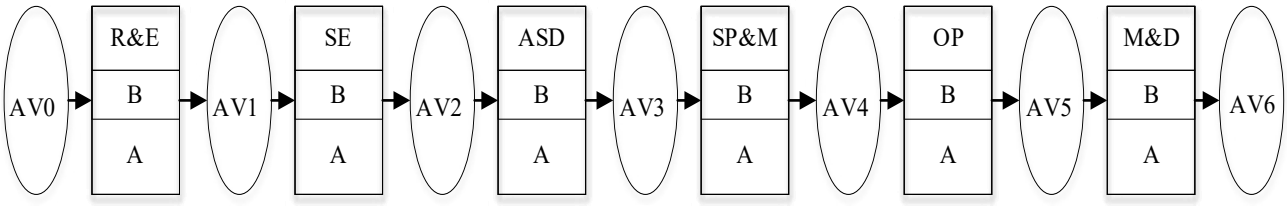

Fig. 2. Software engineering value chain

On fig. 2, the following designations are used:

- AV0 - added value from previous activities (in this case, these are the results of research in fundamental sciences that are not related to software engineering);
- R&E - Research and Education activity - fundamental research and education in software engineering; the main biotic component consist of researchers and teachers, as well as other actors who perform supporting roles, such as trainers, publishers, editors, etc.; abiotic component consist of educational standards, accreditations, universities, journals, conferences, professional organizations, for example, ACM, IEEE;
- AV1 - added value from R&E activities, for example, theories, approaches, principles aimed at the development of software engineering, and engineers, scientists prepared to work in software engineering;
- SE - Software Engineering activity - applied research in software engineering; the main biotic component, these are software engineers, as well as other actors who perform supporting roles, for example, technical and maintenance personnel; the abiotic component, these are, professional websites, magazines, organizations such as Association of Software Professionals , Association for Women in Computing , Python Software Foundation , IACSIT Software Engineering Society , GitHub;
- AV2 - added value from SE activities, such as platforms, tools, APIs, reusable components for use in the processes of creating and maintaining application software;

- ASD - Application Software Development activity - life cycle processes focused on the creation of application software; the main biotic component, these are software developers, as well as other actors who perform supporting roles, for example, the actors of supporting processes, products and tools for creating application software; abiotic component, - resources and materials necessary for the implementation of life cycle processes, standards, documentation;

- AV3 - added value from ASD activity, - software product for the application domain;

- SPM&S - Software Product Marketing and Sale activity - processes aimed at the software product market and sales; the main biotic component these are product managers, market analysts and sales managers; abiotic component these are resources and materials necessary for the delivery of the software product to users;

- AV4 - added value from SPM&S activity - software product sold or delivered to the user;

- Op - Operation activity - use of the software product; the main biotic component, these are users, as a rule, from the application domain and the software product maintenance group, as well as other actors ; abiotic component, these are resources and materials necessary for the use of the software product

- AV5 - added value from Op activity, - consultations, user training, changes made to the product and documentation;

- M&E - Maintenance and Evolution activity - maintenance and evolution of the software product; the main biotic component these are maintainers, domain analysts, reworkers and other actors; abiotic component, these are supported software product, technical resources for maintenance and evolution;

-AV6 - added value from M&E activity, - an evolving software product, parts of a legacy software product that are sent to the circulation of materials to create and maintain a new software product.

The software engineering ecosystems territory. By analogy with the ecosystems of ecology, the landscape metaphor is used for software engineering. An ecosystem is a landscape on which the unity of interacting biotic (living) and abiotic (non-living) components is determined to perform the processes of the corresponding activity. Landscapes can be distinguished by the unity of these components. Unity is determined by focusing on the processes aimed at obtaining the corresponding added value. Landscapes can be local or distributed geographically. Local landscapes are homogeneous environments, whit the same technical, social and cultural values, that correspond to the geography of the given landscape. A distributed landscape, if it occupies geographically different territories, obviously cannot be characterized in this way. The totality of landscapes of all activities of value chain form the software engineering territory.

Diversity of ecosystems. Taking into account the diversity of biotopes in the value chain, we can talk about the diversity of software engineering ecosystems. The landscape of each ecotope within a value chain node can be thought of as being composed of other landscapes and thus defining ecosystems corresponding to the nested landscape. In ecology, this corresponds to the concept of an elementary landscape [4]. The application of this view can be illustrated by the example of ASD activity, taking into account the life cycle processes used to create application software. Considering, for example, the sequential life cycle model, it can be represented as a value chain, where each activity will unfold on the corresponding landscape. Landscape together with biota and abiota will represent the ecosystem. Similarly, incremental, evolutionary, spiral and other life cycle models based on the sequential life cycle model can be considered. The same view can be applied to models of Agile methodology.

Finally, the diversity of software engineering ecosystems can be extended to the engineering that are parts of it. This applies to reverse engineering and empirical software engineering. For example, for reverse engineering, can build a value chain, where each added value will correspond to the knowledge obtained as a result of software analysis activity at the corresponding level of its presentation. Landscape, biota, and abiota each of activity will be ecosystem.

A diversity of activities and biotopes will create a diversity of software engineering ecosystems. We propose to define the software engineering ecosystem using the term “landscape” and the designation of the corresponding activity in the value chain. For example, for the activity «Application Software Development» is defined the ecosystem of Application Software Development landscape.

What about the Software ecosystem?

From the point of view of this work, the factor that plays a fundamental role in the definition of an ecosystem is the landscape. It, in the sense taken here earlier, is absent in the known definitions of the software ecosystem. However, the term “software ecosystem” can be used if software is a landscape. Obviously, then we can talk about some software system, the parts of which, being in relationships and interacting, will represent the abiotic component of the ecosystem. The spatial structure of this abiotic component will be represented by the software landscape, which can be visualized. At the same time, obviously, the software landscape will be is a terrain, that is a biotope, which, strictly speaking, does not contain biota. However, it is also possible to represent the biotic component of the software landscape, if we draw a parallelism between the abiota components of the Software ecology and Environmental Biology (Table 1).

By analogy with ecology [4], there may be tasks of studying the space-time structure of the software ecosystem, information and control flows (algorithms), the principles of evolution and software circulation (Subroutines, Modules (Classes), Megamodules), when legacy software is reused [3].

Table 1. Software Ecology and Environmental Biology Analogies

№	Software Ecology	Environmental Biology
1.	Alphabet	Molecules
2.	Lexemes, simple types	Cells
3.	Operators, complex types	Tissues
4.	Structured operators	Organs
5.	Subroutines	Organs (organisms)
6.	Modules (Classes)	Organs (organisms)
7.	Megamodules	Organs (organisms)
8.	Software life cycle products	Organisms (population)
9.	Software products line (family)	Population
10.	Software of system of systems (ecosystem)	Biogeocenos
11.	Information flows	Energy flows
12.	Control flows (algorithms)	Food chains

Software engineering ecosystem model

Figure 3 presents a model of the software engineering ecosystem. Initially, the space that the ecosystem will occupy is the landscape. Biota is located on the landscape. After that, the landscape can be considered as an ecotope. The biota begins the activity of transforming the ecotope. The ecotope is transformed into a biotope. It is important to determine the nature and results of the biota's activity in transforming an ecotope into a biotope.

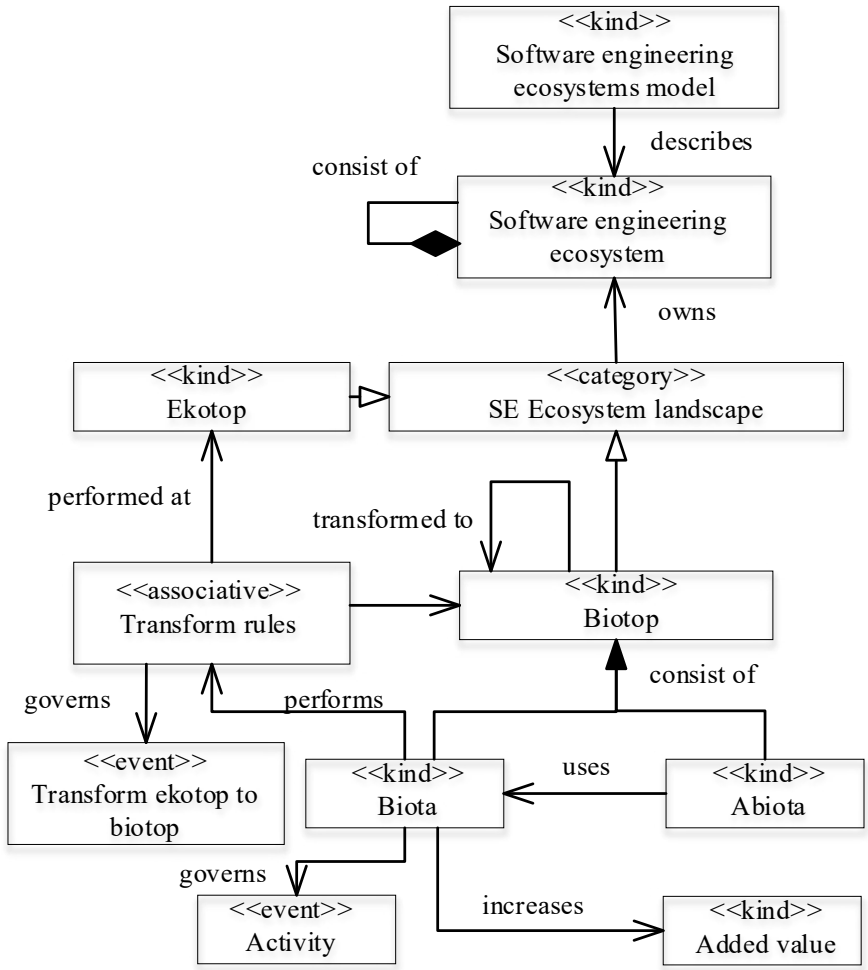

Fig. 3. Software engineering ecosystem model (SE - Software Engineering)

This can be done using of the software engineering culture [9]. The main difference between a biotope and an ecotope is the best conditions for performing an activity. The change of the ecotope conditions is aimed at getting more added value after performing this activity. The means to implement change will be different maturity models

(CMMI, P-CMM), standards, ethics, organizational paradigms, open platforms. For example, in [10] for creating the environment, tools, and activities of DevSecOps as an ecosystem the preparation phase is used. The phase includes understanding the amount of cultural and process change that is required, and identify resources and a strategy that will be transformed the current culture into one that supports DevSecOps principles and behaviors. At the same time, in the future, in order to increase the added value, changes can be carried out in parallel with the biota performing the main activity using the abiotic component of the biotope.

Types of the software engineering ecosystems

Fig. 4 shows the classification of the software engineering ecosystems. According to the nature of the landscape, all ecosystems can be divided into two types - ecosystems of activity landscape and ecosystems of software landscape.

Ecosystems of the first type initially occupy some undeveloped space - an ecotope. Further, ecosystems of the first type is divided into two groups. In the first group, we include ecosystems of landscapes of the value chain (Fig. 2), and in the second group, ecosystems of engineering included in the software engineering.

Ecosystems of the second type is represented by an equipped landscape - software landscape (terrain, biotope). According to the type of software landscape, ecosystems of the second type can be divided into three groups. Software life cycle products (software systems) as ecosystems form the first group. In this case, the Software ecosystem should be considered as a population of organisms and, therefore, as a system of the supraorganismal level. The second group is formed (if it is taken into account) the concept of Individual-based Modeling [11]. Ecosystems are based on individuals and the properties of individuals determine the properties and behavior of the ecosystem as a whole. Therefore, it makes sense to model individuals in the context of a software system as an ecosystem. For example, the software artifact ecosystem [12]. The third group is formed by software of system of systems ecosystems, in which Software life cycle products act as organisms. For example, big data software ecosystem.

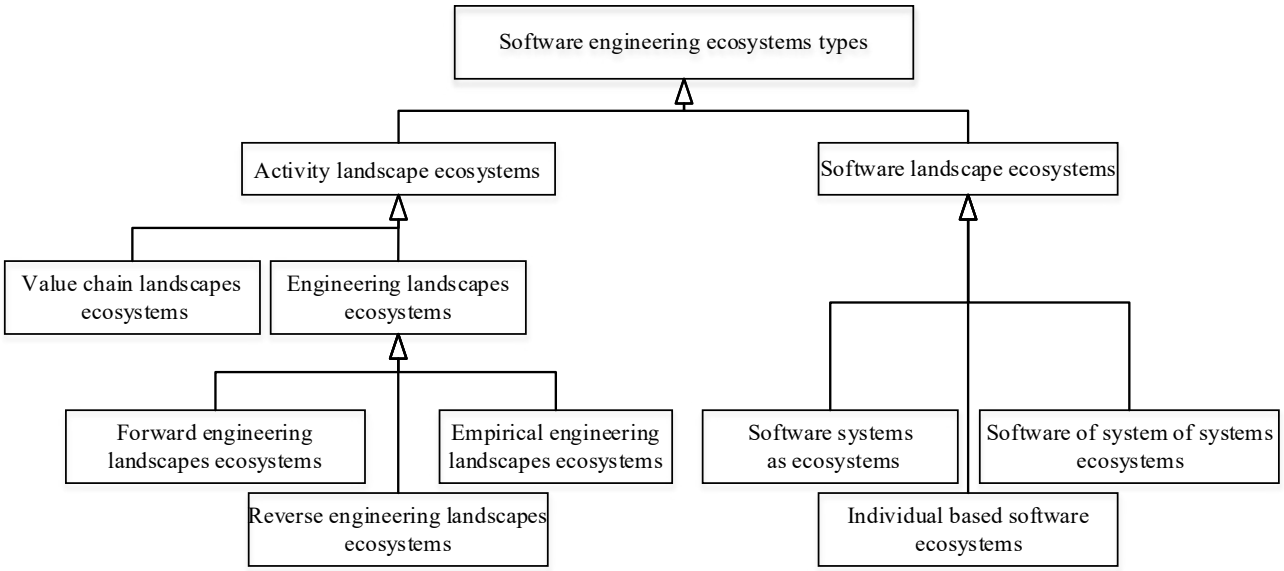

Fig. 4. Types of the software engineering ecosystems

Can note that several or all types of ecosystems can take place in the ecosystems of large companies, for example, IBM, SAP, Microsoft [13].

Related works

The literature on software ecosystems is extensive and presents ecosystem definitions, requirements, models, and case studies. The state of research is reflected in systematic reviews [13 - 16].

In the work [13] stated that out of 90 analyzed works, 40 do not define the software ecosystem, but use definitions from other works. In the remaining 50 papers, four definitions are used with references to the authors (in [15], six definitions are indicated). However, these four definitions also turned out to be so different that the authors of work [13] had to look for something in common in order to formulate one definition. Obviously, this suggests that among researchers there is no consensus on the definition of the term “software ecosystem”. The ambiguous state of affairs is clearly seen from the results of the work [16], which sets the goal of is to develop a common language for the software ecosystem domain. Examining the literature on five topics that relate to the components of software ecosystems, the authors present the following results. There are 46 types of entities found on the topic actors and roles. In addition, in the work [15] identified over 90 roles. At the same time, there is a huge spread of actors in the list, from Researcher to Banks and Investors. On the topic Products and Platforms, the same picture is observed. A total, 27 entities were found, ranging from API to Use case. On the Strategy topic, 21 entities have been identified, and the spread is still large, from the Product lifecycle strategy to Licensing. Finally, on the Boundaries topic, seven entities are identified, and the range is from Abstraction level

to Output. In the work [16], only eight papers were identified that dealt with the boundaries of the software ecosystem. The term Environment is used in three works as a context in which software products, services operate, and ecosystem is a collection of software projects which are developed and which coevolve together in the same environment. [17 – 19].

Analyzing the composition of the entities indicated in [16], one should pay attention not so much to their number and diversity, but most importantly to their disunity. For example, finding the Researcher and Hedger actors or Community driven and Executable components (Products and Platforms topic) in the same ecosystem. It is like finding a lion and a penguin in the same biological ecosystem. The same can be said about other entities. Obviously, this indicates that the ecosystem or its boundaries are incorrectly defined.

In the result of the analysis of related works, it has been suggested that the term software ecosystems is now actually used to refer to a wider range of ecosystems, which are actually software engineering ecosystems. Therefore, the purpose of our work was to propose such a definition of software engineering ecosystems that would allow us to avoid the existing ambiguity.

Conclusion

Based on the hypothesis that the term "software ecosystems" is used to refer to a wider range of ecosystems, which are actually software engineering ecosystems, the goal was to proposed a base for defining software engineering ecosystems. As such, a base, by analogy with biological ecosystems, the concepts of landscape and value chains are proposed. Based on these concepts, the existing variety of software engineering ecosystems is pointed out, a model of the software engineering ecosystems and the classification of the software engineering ecosystems are proposed.

This paper is a continuation of the author's works on the topic [3, 12, 20 - 23]. In future works, in order to confirm the theses of the paper, it is undoubtedly necessary to conduct a Case study, perhaps on the examples of companies like Microsoft, IBM, which have all types of the software engineering ecosystems on their territory.

References

1. Bourque P., Fairley R.E., eds., Guide to the Software Engineering Body of Knowledge, ver. 3.0, IEEE CS, 2014; www.swebok.org.
2. Nguyen T. N., The Ecology of Software: A Framework for the Investigation of Business-IT Integration Issues and Trends of Information Technology Management in Contemporary Organizations, the proceedings of the Information Resources Management Association International Conference, 2002
3. Sydorov N.A.Software ecology, Software engineering, №1, 2010 (in Ukrainian).
4. Odum E P. Fundamentals of Ecology.. Saunders, Philadelphia, 1959.
5. Fundamentals of forest biogeocenosis, edited by V.N.Sukachev, M. Science, 1964 (in Russian)
6. Perry D. E., Kaiser G. E., Models of Software Development Environments, IEEE Transactions on Software Engineering, 1991. – Vol. 17, N. 3. – P.283-295
7. Balzer M., Noack A., Deussen O. Software Landscapes: Visualizing the Structure of Large Software Systems, IEEE TCVG Symposium on Visualization (VisSym), May 19-21, 2004, Konstanz
8. Biffl, S. Aurum A., Boehm B., Value-Based Software Engineering, Springer, 2006, 398p.
9. Wiegers K., Creating a Software Engineering Culture, Dorset House Publishing, 1996
10. Guide to Implementing DevSecOps for a System of Systems in Highly Regulated Environments, TECHNICAL REPORT CMU/SEI-2020-TR-002, April 2020, 111p
11. V. Grimm, S. F. Railsback, Individual-based Modeling and Ecology, Princeton University Press, 1999, 429 p.
12. Sydorov N., Programming Style As An Artefact Of A Software Artefacts Ecosystem, Advances in Computer Science for Engineering and Education IV, Springer
13. Manikas K., Hansen, K.M.: Software ecosystems-a systematic literature review. J. Syst. Softw. 86(5), 1294–1306 (2013)
14. García-Holgado A., García-Peñalvo F. J., Mapping the systematic literature studies about software ecosystems, Proceedings TEEM'18. Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality (Salamanca, Spain, October 24th-26th, 2018), (pp. 910-918). New York, NY, USA: ACM. doi: 10.1145/3284179.3284330
15. Suortti S., The Role of Software Platform and Actors in Software Ecosystems: A Case Study in Agriculture, Aalto University, 2017
16. Wouters J., Ritmeester J. R., Carlsen A. W., A SECO Meta-model A Common Vocabulary of the SECO Research Domain, Springer Nature Switzerland AG, 2019
17. Boucharas, V., Jansen, S., Brinkkemper, S.: Formalizing software ecosystem modeling. In: Proceedings of the 1st International Workshop on Open Component Ecosystems, IWOCE 2009, pp. 41–50, August 2009
18. Lungu, M. F. Reverse Engineering Software Ecosystems, Doctoral Dissertation submitted to the Faculty of Informatics of the University of Lugano, September 2009
19. Knodel J., Manikas K., Towards a Typification of Software Ecosystems, Software Business 6th International Conference, ICSOB 2015 Braga, Portugal, June 10–12, 2015 Proceedings, P77-82
20. Луцкий М., Сидоров Н., Программное обеспечение – экологический подход к исследованиям, Natural and Artificial Intelligence. – ITHEA.–2010.–Sofia.–Bulgaria.–P 181-189
21. Sydorov N.A, Sydorova N.N Sydorov E.N, Description model of programming style ecosystem, Problems in programming, 2-3, 2020, P 74-81
22. Sydorov N.A. Toward software artifacts ecosystem, Problems in programming, 4, 2020, P 110-120
23. Sydorov N.A. Programming Style as an Artefact of a Software Artefacts Ecosystem, International Conference on Computer Science, Engineering and Education Applications, Springer, Cham, 2021, Kiev, P 244-255

Література

1. Bourque P., Fairley R.E., eds., Guide to the Software Engineering Body of Knowledge, ver. 3.0, IEEE CS, 2014; www.swebok.org.
2. Nguyen T. N., The Ecology of Software: A Framework for the Investigation of Business-IT Integration Issues and Trends of Information Technology Management in Contemporary Organizations, the proceedings of the Information Resources Management Association International Conference, 2002
3. Сидоров Н.А.Екологія програмного забезпечення, Инженерія програмного забезпечення №1 2010
4. Odum E P. Fundamentals of Ecology.. Saunders, Philadelphia, 1959.
5. Основы лесной биогеоценологии под редакцией В. Н. Сукачева, М. Наука, 1964
6. Perry D. E., Kaiser G. E., Models of Software Development Environments, IEEE Transactions on Software Engineering, 1991. – Vol. 17, N. 3. – P.283-295

7. Balzer M., Noack A., Deussen O. Software Landscapes: Visualizing the Structure of Large Software Systems, IEEE TCVG Symposium on Visualization (VisSym), May 19-21, 2004, Konstanz
8. Biffl, S. Aurum A., Boehm B., Value-Based Software Engineering, Springer, 2006, 398p.
9. Wiegers K., Creating a Software Engineering Culture, Dorset House Publishing, 1996
10. Guide to Implementing DevSecOps for a System of Systems in Highly Regulated Environments, TECHNICAL REPORT CMU/SEI-2020-TR-002, April 2020, 111p
11. V. Grimm, S. F. Railsback, Individual-based Modeling and Ecology, Princeton University Press, 1999, 429 p.
12. Sydorov N., Programming Style As An Artefact Of A Software Artefacts Ecosystem, Advances in Computer Science for Engineering and Education IV, Springer
13. Manikas K., Hansen, K.M.: Software ecosystems-a systematic literature review. J. Syst. Softw. 86(5), 1294–1306 (2013)
14. García-Holgado A., García-Peñalvo F. J., Mapping the systematic literature studies about software ecosystems, Proceedings TEEM'18. Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality (Salamanca, Spain, October 24th-26th, 2018), (pp. 910-918). New York, NY, USA: ACM. doi: 10.1145/3284179.3284330
15. Suortti S., The Role of Software Platform and Actors in Software Ecosystems: A Case Study in Agriculture, Aalto University, 2017
16. Wouters J., Ritmeester J. R., Carlsen A. W., A SECO Meta-model A Common Vocabulary of the SECO Research Domain, Springer Nature Switzerland AG, 2019
17. Boucharas, V., Jansen, S., Brinkkemper, S.: Formalizing software ecosystem modeling. In: Proceedings of the 1st International Workshop on Open Component Ecosystems, IWOCE 2009, pp. 41–50, August 2009
18. Lungu, M. F. Reverse Engineering Software Ecosystems, Doctoral Dissertation submitted to the Faculty of Informatics of the University of Lugano, September 2009
19. Knodel J., Manikas K., Towards a Typification of Software Ecosystems, Software Business 6th International Conference, ICSOB 2015 Braga, Portugal, June 10–12, 2015 Proceedings, P77-82
20. Луцкий М., Сидоров Н., Программное обеспечение – экологический подход к исследованиям, Natural and Artificial Intelligence. – ITHEA.–2010.–Sofia.–Bulgaria.–P 181-189
21. Sydorov N.A, Sydorova N.N Sydorov E.N, Description model of programming style ecosystem, Problems in programming, 2-3, 2020, P 74-81
22. Sydorov N.A. Toward software artifacts ecosystem, Problems in programming, 4, 2020, P 110-120
23. Sydorov N.A. Programming Style as an Artefact of a Software Artefacts Ecosystem, International Conference on Computer Science, Engineering and Education Applications, Springer, Cham, 2021, Kiev, P 244-255

Received 17.08.2022

Про автора:

Сидоров Микола Олександрович,
доктор технічних наук,
професор.
Кількість наукових публікацій у
українських виданнях – 140.
Кількість наукових публікацій у
зарубіжних виданнях – 22.
http://orcid.org/0000-0002-3794-780X

Місце роботи автора:

Національний Технічний Університет України
«Київський політехнічний інститут імені Ігоря Сікорського»,
факультет інформатики та обчислювальної техніки,
кафедра інформатики та програмної інженерії, ІПІ, професор.
02000, Київ,
вул. Політехнічна, 41.
Моб. тел.: 067 7980361.
E-mail: nyksydorov@gmail/com

Прізвища та ім'я авторів і назва доповіді англійською мовою:
Sydorov Nikolay
Toward software engineering ecosystems definition

Прізвища та ім'я авторів і назва доповіді українською мовою:
Сидоров Микола Олександрович
Щодо визначень екосистем інженерії програмного забезпечення