https://orcid.org/0000-0002-4168-7190

UDC 623.764

P.S. SAPATY*

SELF-RECOVERING NETWORKS UNDER SPATIAL GRASP TECHNOLOGY

*Institute of Mathematical Machines and Systems Problems of the National Academy of Sciences of Ukraine, Kyiv, Ukraine

Анотація. *Самовідновлення є надзвичайно важливою властивістю великих систем на національному, міжнародному та навіть глобальному світовому рівнях. Оскільки будь-які системи, особливо великі та розподілені, часто можна представити у вигляді мережі з вузлами, їх компонентами та зв'язками як комунікаціями між ними, відновлення будь-яких систем можна розглядати насамперед як відновлення їх мережевих структур. І для цього відновлення часто може знадобитися використання здебільше внутрішніх системних ресурсів із мінімальним зовнішнім втручанням, доповненням або контролем, оскільки в інших системах також можуть виникнути різні проблеми, і вони не зможуть ділитися своїми ресурсами. У статті досліджено та детально показано, як розроблена модель та Технологія просторового захоплення (ТПЗ) з її рекурсивною Мовою просторового захоплення (МПЗ) можуть організувати розподілені мережі будь-яких обсягів і топологій, щоб бути справді самовідновлюваними та фактично «безсмертними». Стаття пропонує універсальне рішення, де всі мережеві вузли, будучи потенційно активними, можуть спільно проводити аналіз мережі на корисну глибину від кожного з них та обмінюватися результатами, згодом надаючи кожному вузлу повний опис всієї мережі, а потім використовувати цей опис для колективного відновлення всієї мережі від будь-яких пошкоджень, якщо хоча б один вузол ще залишається живим. Це рішення використовує просторово керовану унікальну гнучкість ТПЗ, подібну до супервірусу, та самовідтворення сценаріїв МПЗ, які вільно переміщуються між вузлами мережі, використовуючи лише локальні зв'язки між ними. Стаття також містить практичні рекомендації щодо використання запропонованого рішення для самовідновлення дуже великих мереж, а також, як додатково залучати інші критичні інфраструктури у разі складних криз і проблем у соціальних системах.*

Ключові слова: *критичні інфраструктури, великі розподілені мережі, мережеві загрози, самоаналіз і самовідновлення, Технологія просторового захоплення, Мова просторового захоплення, кооперативна поведінка.*

Abstract. *Self-recovery, often mentioned as self-healing and remediation, is an extremely important superpower-like feature of large systems on national, international, up to the global level. As any systems, especially large and distributed, can often be represented in a network form, with nodes as their components and links as communications in between, the recovery of any systems may be considered first of all as the recovery of their network structures. And this recovery may often need using mostly internal system resources with minimum external intervention, supplement, or control, as other systems may have problems too and be unable to share their resources. The paper investigates and shows in detail how the developed Spatial Grasp Model and Technology with its recursive Spatial Grasp Language can organize distributed networks of any volumes and topologies to behave in a really self-healing, self-repairing, actually "immortal" manner. It offers a universal solution where all networked nodes, being potentially active, can cooperatively provide network analysis to the reasonable depth from each of them, share the results by subsequently supplying each node with the full description of the whole network, and then use this description to collectively restore the whole network from any damages if at least a single node still remains alive. This solution uses spatially controlled unique supervirus-like flexibility and self-replication of SGL scenarios, freely migrating between network nodes using only local communications among them. The paper also provides practical recommendations of how to use the offered self-recovery solution for huge networks, and shows how to additionally involve other critical infrastructures in case of complex crises and problems in social systems.*

Keywords: *critical infrastructures, large distributed networks, network threats, self-analysis and self-recovery, Spatial Grasp Technology, Spatial Grasp Language, cooperative behaviour.*

DOI: 10.34121/1028-9763-2025-1-32-41

1. Introduction

Practically any existing or imaginable system, especially large and distributed, can be properly represented in a graph or network form, with nodes reflecting its parts or components and arcs or edges reflecting physical or virtual links and communications between them. Any disruptions or damages in distributed systems, witnessed every day and even worldwide, are naturally directing us to inspect, first of all, their network representations, analysis of which may be the basic step towards healing and recovery. To hint at what is going on in the area of network analysis and recovery, these publications may be initially recommended [1–6]. For example, the book [1] provides detailed information on protecting and restoring communication networks. It describes specific techniques that work at each layer of the networking hierarchy, beginning with incisive introduction that defines the field of network protection and restoration, and then explaining everything needed to know about the relevant protocols. It shows how to implement protection and recovery techniques, and how to evaluate these techniques in relation to one another. More on self-healing networks can be found in [7–14] and on typical network threats and risks in [15].

The aim of this paper is to describe and explain in detail a universal approach allowing arbitrary large and complex distributed networks to become self-analyzing, self-updating, self-healing, and self-recovering from any damages and disruptions, and to investigate the potential capability of the developed Spatial Grasp Model and Technology (SGT) and its basic Spatial Grasp Language (SGL) [16–26] to provide these important features.

The rest of the paper, which details and concretizes our recent works [27–29] on self-healing and self-recovering networks and systems, is organized as follows.

In Section 2, a summary on SGT and SGL is provided. Section 3 shows a network example with named nodes and links to be used in the following sections, where nodes are also supposed to have unique coordinates. Section 4 shows how to copy the full network topology with the node and link details by starting from a single node and also distribute this full description to all network nodes. Section 5 shows how to make the solution of the previous chapter more advanced by allowing the repeated network copying independently from any nodes operating cooperatively and in parallel. Section 6 shows how to create the whole network by its obtained description, starting from a single node. Section 7 further develops the previous solution by allowing simultaneous network creation or recovery when starting from many nodes in a cooperative manner. Section 8 effectively combines advanced copying and creation solutions of Sections 5 and 7 into an integral analysis-copying-creation procedure operating simultaneously, repetitively and cooperatively from all network nodes, actually converting the whole network, which may also change its parameters, into a fully autonomous and indestructible object. Section 9 shows how to make the solution of the previous section more practical for huge distributed networks. Section 10 also demonstrates that practical recovery solutions for real networks may need additional engagement of other social infrastructures. Section 11 concludes the paper, summarizing its achievements and hinting at further plans in this area. References include reviewed publications on the network recovery and existing sources on SGT.

Figure 1 — Parallel recursive world coverage with SGT

2. Spatial Grasp Technology summary

Within Spatial Grasp Technology (SGT), a high-level scenario in Spatial Grasp Language (SGL)

self-spreads and matches distributed environments in parallel wave-like mode, as symbolically shown in Fig. 1 (sufficient details on this paradigm can be found in [16–26]). Such propagation can result in returning and analyzing the reached states and data, which can be remote, also in further waves from the initial and obtained points, with this forward-backward world coverage and "grasping" arbitrarily hierarchical and recursive.

SGL top level organization can be expressed just in a single string-formula mode:

grasp → *constant* | *variable* | *rule* ({ *grasp*, })

An SGL scenario, or *grasp*, initially applied in some point (or points) of distributed spaces, can be a *constant* directly providing the result, or a stationary or mobile *variable* with the content obtained from this or other space points. It can also be a *rule* expressing certain ordering, action, or control over a set of operands, generally as grasps too. More on the sense of these basic elements as follows:

constant → *information* | *matter* | *custom* | *special* | *grasp*
variable → *global* | *heritable* | *frontal* | *nodal* | *environmental* | *grasp*
rule → *advancement* | *feedback* | *branching* | *acting* | *sensing* | *context* | *grasp*

The constants, variables, and rules, if needed, can also be represented by arbitrary spatial structures, i.e. as grasps again.

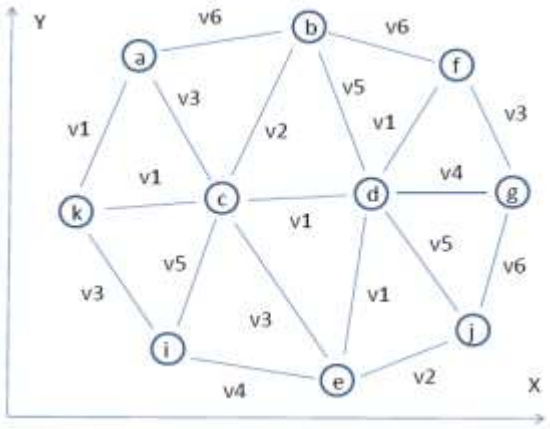

Figure 2 — Representation of a distributed network

SGL can express and provide effective solutions in large distributed networked systems in a simple and compact way. Communicating SGL interpreters can be in arbitrary numbers (up to thousands and millions, if required) integrated with other systems and communications, forming altogether *powerful spatial engines* operating without central management or control.

3. Network example

An example of a network is shown in Fig. 2, which may have an arbitraily number of named nodes connected by any number of named links, forming altogether any structures and topologies, with nodes also having individual and unique addresses in physical or virtual spaces.

4. Copying full topology from a single node, with the result in all nodes

This can be done in a self-evolving spanning tree mode, starting from any single graph node, as follows:

```
hop_node(any); frontal(Topf, Locf);
Topf = repeat_first(
blind(NAME & unit(jump_links(all); LINK & NAME)),
hop_links(all));
```

```
Locf = repeat_first(
blind(NAME & WHERE), hop_links(all));
repeat_first(
Top = Topf; Loc = Locf; hop_links(all))
```

This scenario, starting from a single node, uses top-down network coverage in acyclic spanning tree mode (using `repeat_first` repetition rule and moving only through the network links between nodes), which is picking up from each node its name and a list of named neighbouring nodes together with named links to them. After that, in a bottom-up feedback mode, it collects the data obtained at all nodes as a full network description and assigns it to the frontal variable `Topf` in the starting node. In a similar combined top-down and feedback bottom-up mode, a combination of node names and their individual location-addresses is collected for all network nodes and assigned to the frontal variable `Locf` in the starting node too. After this, the collected data on the whole network in `Topf` and `Locf` is distributed to all network nodes in the top-down acyclic mode again and assigned in them to the related variables `Top` and `Loc` which, as stationary ones, will be preserving in them the whole network description permanently. This multiple top-down and bottom-up network coverage process is depicted in Fig. 3.

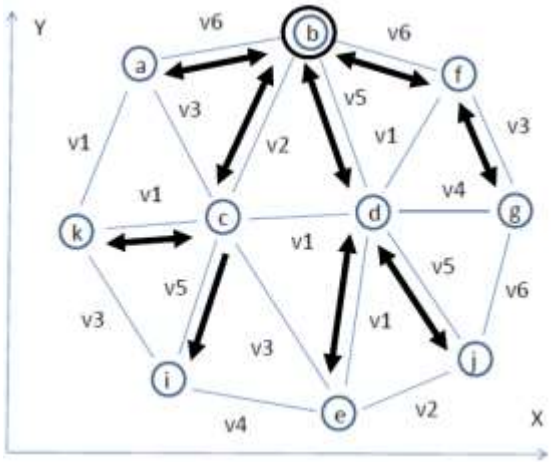

Figure 3 — Copying a distributed network starting from a single node

Collected full network description accumulated in variables `Top` and `Loc` may look as follows:

```
Top = (a:(v1:k, v3:c, v6:b), b:(v6:a, v2:c, v5:d, v6:f), f:(v6:b,
v1:d, v3:g), k:(v1:a, v1:c, v3:i), c:(v1:k, v3:a, v2:b, v1::d,
v3:e, v5:i),
d:(v1:c, v5:b, v1:f, v4:g, v5:j, v1:e), g:(v4:d, v3:f, v6:j),
i:(v3:k, v5:c, v4:e), e:(v4:I, v3:c, v1:d, v2:j), j:(v2:e, v5:d,
v6:g))

Loc = (a:Ca, b:Cb, c:Cc, d:Cd, e:Ce, f:Cf, g:Cg, I:Ci, j:Cj,
k:Ck)
```

5. Repeated updating the network copy from each node

We can modify the network copying scenario from the previous section not only to cover and copy the whole network and then distribute its full description to all network nodes, but also add to this distribution the description of the full scenario itself, and then activate it in all the nodes reached. This will be making all the nodes capable of navigating and copying the whole network independently, in any order, in parallel, and at any time, with reasonable delays between different repetitions. Also, the modified scenario, when distributing the obtained full network description from a node to all other nodes, may discover that these nodes already have the network description brought here from other nodes. In this case, it simply updates this network description version with the new one, which may be larger or more correct, and this may be extremely useful if the previous description may happen not to be full, say, as different nodes cannot cover the whole network for some communications reasons. Or this coverage was limited to some practically rec-

ommended or allowed depth, in case of very large networks. This will allow us to steadily accumulate full network description in all nodes, including the possible appearance of very new network nodes and their links during this continuous global copying cooperative process.

The resultant global copying scenario is represented as a stationary procedure named and run as `Copy`, which before the distribution in the network is represented by the frontal variable `Copyf` and then, when reaching different nodes, is settled as `Copy` again in these nodes and regularly activated, as follows:

```
hop_node(any);
Copy =
{frontal(Topf, Locf, Copyf);
 repeat
 stay(
  Topf = repeat_first(
  blind(NAME & unit(jump_links(all); LINK & NAME)),
  hop_links(all));
  Locf = repeat_first(
  blind(NAME & WHERE), hop_links(all));
  Copyf = Copy;
  repeat_first(
  update(Top, Topf); update(Loc, Locf);
  if(Copy == nil, free_run(Copy = Copyf));
  hop_links(all)));
 sleep(delay))};
run(Copy)
```

6. Creating full topology starting from a single node

Creating the same network topology starting from some `Start` node having information about the whole network in frontal variables `Topf` and `Locf` and using spatially evolving top-down spanning tree mode may be accomplished by the following scenario (see also Fig. 4):

```
frontal(Topf = …, Locf = …); nodal(Start) = …;
create_node(Start, Locf:Start);
repeat(
 split(Topf:NAME);
 or_seq(
  (empty(Locf:VAL[2]);
  create(link(VAL[1], node(VAL[2], Locf:VAL[2])),
  (NAME > VAL[2];
  quit_linkup(VAL[1], node(VAL[2], Locf:VAL[2])))))
```

As the network-forming spanning tree process develops asynchronously in physical space, the competition between the already existing nodes for creating the same next nodes is resolved by first testing whether this node's physical position is still empty, and if yes, creating this node with the link to it. Otherwise, trying to create a link to the already existing node if such link does not exist yet, resolving competition between the neighbouring nodes for doing the same by comparing their names as values before the link creation, and then stop at this point of space after the successful or unsuccessful creation of this link.

7. Simultaneous network creation-recovery from different nodes

In the previous section, we described the creation of a full networking topology by starting from a single node, in which the whole network description existed. We can modify this scenario in such a way that the creation of the network may start simultaneously from different nodes, with this top-down spanning tree processes terminating as soon as the next nodes to be created already exist, being obtained by other branches of the same spanning tree process or by spanning tree processes started in other nodes, as shown in Fig. 5 (when originally starting from nodes `k` and `g`).

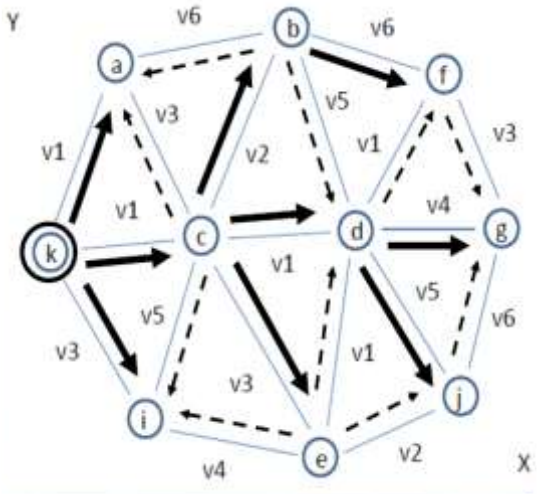

Figure 4 — Creating a distributed network starting from a single node

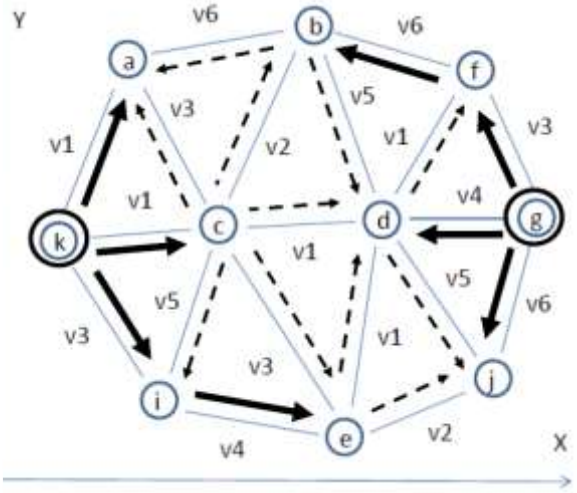

Figure 5 — Simultaneous network creation starting from different nodes

This idea can be further improved by supplying each newly created node with the whole network creation scenario and the full network topology description in frontal variables `Topf` and `Locf`, after which these new nodes become capable of starting this network creation process autonomously too. This may be extremely important if nodes and links of the network can happen to be arbitrarily destroyed, and the whole network can be restored if some nodes still remain alive, even a single one. The resultant scenario represented as procedure `Rec` first activated in all still existing nodes and then transferred to the newly created nodes in frontal variable `Recf`, stored as stationary `Rec`, and regularly activated may be as follows:

```
hop_nodes(all);
Rec =
{frontal(Topf = ..., Locf = ..., Recf);
 Recf = Rec;
 repeat(
 stay_repeat(
  split(Topf:NAME);
  or_seq(
  (empty(Locf:VAL[2]);
   create(link(VAL[1], node(VAL[2], Locf:VAL[2]));
   Top = Topf; Loc = Locf); free_run(Rec = Recf)),
  (no(hop(link(VAL[1], node(VAL[2]));
   quit_linkup(VAL[1], node(VAL[2], Locf:VAL[2])))));
 sleep(delay))};
```

```
run(Rec)
```

8. Combined universal self-copying and self-recovering solution

In Section 5, we described a universal network copying scenario that is regularly operating in collective activities simultaneously emanating from all the existing nodes, which results in full network description in all the nodes even if there may happen communication problems between them or the network can change at the same time in volume and space. We named such universal procedure stationed in all the nodes as `Copy`, which can be exchanged as a whole between different nodes in the frontal variable `Copyf`, while converting in the reached nodes into stationary `Copy` again.

In Section 7, we described a universal procedure that with a full network description can start from a single or simultaneously multiple nodes and create the full network in the distributed space or just repair it after any damages, even if a single node still remains alive. We have named this universal network creation-recovery procedure as `Rec` as stationary in nodes, or `Recf` when exchanged between nodes and installed in destinations as `Rec` again.

Having analyzed in detail the potential capabilities of these `Copy` and `Rec` universal procedures, we are happy to offer now their *effective deep integration*, which will enable continuous both updating-copying and creation-recovery of networks with any volumes, topologies, and spatial distribution, also taking into account that these networks can change their parameters at any time. Such integrated procedure named `CopyRec` when stationary in nodes may look as follows, being transferred, if needed, between nodes in the frontal variable `CopyRecf`:

```
hop_nodes(all);
CopyRec =
{frontal(Topf, Locf, CopRecf);
 CopyRecf = CopyRec;
repeat(
 stay(
  Topf = repeat_first(
  blind(NAME & unit(jump_links(all); LINK & NAME)),
  hop_links(all));
  Locf = repeat_first(
   blind(NAME & WHERE), hop_links(all));
  repeat_first(
   update(Top, Topf); update(Loc, Locf);
   if(CopRec == nil, free_run(CopyRec = CopyRecf));
   hop_links(all)));
 sleep(delay1);
 stay_repeat(
  split(Topf:NAME);
  or_seq(
  (empty(Locf:VAL[2]);
   create(link(VAL[1], node(VAL[2], Locf:VAL[2]));
   Top = Topf; Loc = Locf; free_run(CopyRec = CopyRecf)),
  (no(hop(link(VAL[1], node(VAL[2]));
   quit_linkup(VAL[1], node(VAL[2], Locf:VAL[2])))));
 sleep(delay2))};
run(CopRec)
```

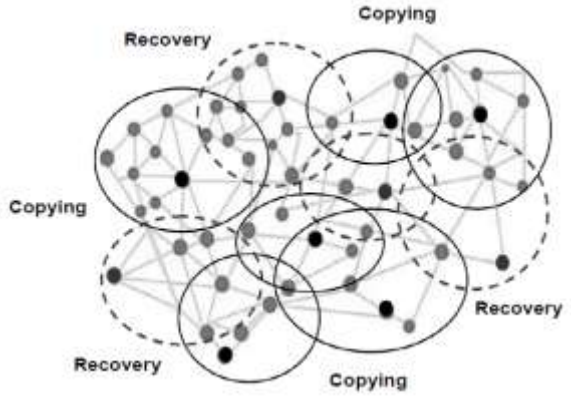

Figure 6 — Using the solution for large dynamic networks

9. Self-recovery of large networks

A symbolic depiction of the activity of many nodes executing `CopyRec` procedure for large networks is shown in Fig. 6, where each node emitting copying or recovery activity may happen to limit the depth of its parallel network coverage to a certain reasonable depth rather than the whole network. But repeating parallel and cooperative activity of nodes with exchange of results obtained by each of them may be sufficient to keep the whole network, whatever large, complex, and spatially distributed it might be, to autonomously withstand any damages and destruction that may happen in it.

10. Using additional infrastructures for practical recovery

As shown in Fig. 7, after discovering road damage and repairing the road network virtually by the `CopyRec` procedure described above, the *Transport*, *Terrain*, *Engineering*, *Employment*, and *Economy* infrastructures (possibly others too) have also been engaged for restoring the damaged crossroad 12 together with its roads to other crosses. This repair was, however, practically accomplished only to crossroads 8, 13, 4, and 6, whereas other missing roads to crosses 2 and 3 appeared too difficult and costly to be restored now, with their renewed consideration being left for the future.

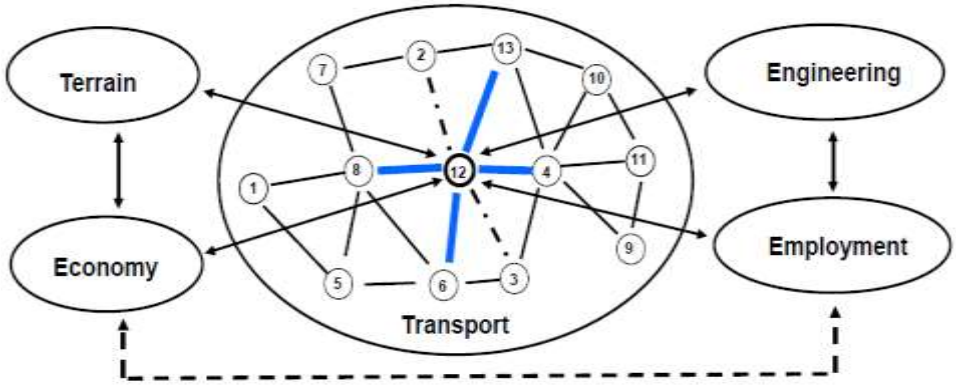

Figure 7 — Cooperative engagement of different infrastructure for road network recovery

11. Conclusions

We have described in detail a universal solution of how to convert an arbitrary distributed network into a constantly self-analyzing, self-copying, and self-recovering system after any disruptions and damages, which can be effectively done by cooperative activity of its nodes, each of which can navigate the network to a certain depth. And all this can be accomplished by using only existing direct communication lines between individual nodes without any external communications, management, and control. This universal solution was obtained stepwise by first offering a solution for self-copying and updating the network structure with storing its resultant description in all existing nodes and then showing how to use this information for repairing the whole network after any damages. The resultant combined self-analysis copying-recovery solution was presented in high-level Spatial Grasp Language, whose parallel scenarios, like a power-

ful self-controlled and self-replicating super-virus, are self-spreading in a wave-like mode in distributed systems and organizing their collective behaviour in a compact and clear way.

Taking into account the experience from previous publications [17–26], this may be up to a hundred times shorter and proportionally simpler than in traditional languages like C or Java. And as already mentioned in Section 1, this paper further develops, details, and concretizes our pioneering work on self-healing and self-recovering systems [27–29]; it also effectively uses our techniques for dealing with large network-represented distributed systems [26].

Further investigation of practical application of the developed self-recovery technique for very different types of systems is on the agenda, as well as an international project on self-healing and self-recovery after crises and disasters, which are frequently emerging worldwide.

REFERENCES

1. Vasseur J.-P., Pickavet M. et al. Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP and MPLS (ISSN). Morgan Kaufmann. 1st edition. 2004. August 19. 544 p.
2. Tootaghaj D.Z., Bartolini N. et al. On Progressive Network Recovery From Massive Failures Under Uncertainty. *IEEE Transactions on Network and Service Management.* 2019. Vol. 16, Issue 1. March. DOI: https://doi.org/10.1109/TNSM.2018.2880155. URL: https://ieeexplore.ieee.org/document/8527547.
3. Awoyemi B.S., Alfa A.S. et al. Network Restoration for Next-Generation Communication and Computing Networks. *Journal of Computer Networks and Communications*. John Wiley & Sons Ltd. 2018. April 3. DOI: https://doi.org/10.1155/2018/4134878. URL: https://onlinelibrary.wiley.com/doi/10.1155/2018/413 4878.
4. Kellerman R. What is Network Recovery as a Service (NRaaS) and why do I need it? URL: https://stage2data.com/what-is-network-recovery-as-a-service-nraas-and-why-do-i-need-it/.
5. What Is Disaster Recovery? CISCO. URL: https://www.cisco.com/c/en/us/products/ security/what-is-disaster-recovery.html.
6. What Is a Self-Healing Network? Definition & How It Works, Nile, AI Networking. URL: https://nilesecure.com/ai-networking/what-is-a-self-healing-network-definition-how-it-works.
7. Gu X., Jiang N. et al. Self-healing Control Technology for Distribution Networks. Wiley; 1st edition. 2017. February 7. 204 p.
8. Chaudhry J. A. Self-Healing Systems and Wireless Networks Management. CRC Press; 1st edition. 2013. October 21. 182 p.
9. Ramiro J., Hamied K. Self-Organizing Networks: Self-Planning, Self-Optimization and Self-Healing for GSM, UMTS and LTE. Wiley; 1st edition. 2011. October 27. 321 p.
10. Zacchello M. Network, Heal Thyself: On Self-Healing Networks. How automation, AI and ML are delivering more intelligent networks that can predict, repair and maintain themselves. 2023. August 29. URL: https://blog.equinix.com/ blog/2023/08/29/network-heal-thyself-on-self-healing-networks/.
11. Chaban M.A.V. How self-healing networks help keep the digital world stable and secure. 2021. May 6. URL: https://www.ibm.com/blog/self-healing-networks-stable-secure-digital-5g-world/.
12. Burger D. What Are Self-Healing Networks? 2019. May 29. URL: https://restechtoday. com/self-healing-networks/.
13. Scala A., Morone F., Makse H. Self-healing networks. In book: Safety and Reliability – Safe Societies in a Changing World. Taylor & Francis Group, London. 2018. URL: https://www.researchgate.net/publication/329697742_Self-healing_networks.
14. Elliott C., Heile B. Self-organizing, self-healing wireless networks. 2000 IEEE Aerospace Conference. Proc. (Cat. No.00TH8484), 2000. 25–25 March. URL: https://ieeexplore.ieee.org/document/879383.
15. Top Network Threats & Risks (And How To Protect Yourself). URL: https://nilesecure.com/network-security/top-network-threats-risks-and-how-to-protect-yourself.
16. Sapaty P.S. A distributed processing system, European Patent N 0389655. Publ. 10.11.93, European Patent Office.
17. Sapaty P.S. Mobile Processing in Distributed and Open Environments. New York: John Wiley & Sons, 1999. 410 p.
18. Sapaty P.S. Ruling Distributed Dynamic Worlds. New York: John Wiley & Sons, 2005. 255 p.

19. Sapaty P.S. Managing Distributed Dynamic Systems with Spatial Grasp Technology. Springer, 2017. 284 p.
20. Sapaty P.S. Holistic Analysis and Management of Distributed Social Systems. Springer, 2018. 234 p.
21. Sapaty P.S. Complexity in International Security: A Holistic Spatial Approach. Emerald Publishing, 2019. 160 p.
22. Sapaty P.S. Symbiosis of Real and Simulated Worlds under Spatial Grasp Technology. Springer, 2021. 251 p.
23. Sapaty P.S. Spatial Grasp as a Model for Space-based Control and Management Systems. CRC Press, 2022. 280 p.
24. Sapaty P.S. The Spatial Grasp Model: Applications and Investigations of Distributed Dynamic Worlds. Emerald Publishing, 2023. 184 p.
25. Sapaty P.S. Providing Integrity, Awareness, and Consciousness in Distributed Dynamic Systems. CRC Press, 2024. 177 p.
26. Sapaty P.S. Spatial Networking in the United Physical, Virtual, and Mental World. Springer, 2024. 443 p.
27. Sapaty P.S. Self-Recovering Infrastructures and Networks under Spatial Grasp Paradigm, in Proceedings of the 7th International Scientific Conference «Theoretical Hypotheses and Empirical results». Oslo, Norway, 2024. July 11–12. DOI: https://doi.org/10.5281/ zenodo.12741275. URL: https://ojs.publisher.agency/index.php/ THIR /issue/view/91/222.
28. Sapaty P.S. Self-Recovering Systems under Spatial Grasp Technology. *Crisis, Stress, and Human Resilience: An International Journal*. International Critical Incident Stress Foundation, Inc. Dec. 2024 (in print). URL: https://www.crisisjournal.org/article/126490-self-recovering-systems-under-spatial-grasp-technology.
29. Sapaty P.S. Self-Healing and Self-Recovering Systems under the Spatial Grasp Model. CRC Press, 2025 (in print).

Стаття надійшла до редакції 12.02.2025