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

UDC 623.764

P.S. SAPATY[*]

PROVIDING DISTRIBUTED SYSTEM INTEGRITY UNDER SPATIAL GRASP TECHNOLOGY

[*]Institute of Mathematical Machines and Systems Problems of the NAS of Ukraine, Kyiv, Ukraine

Анотація. *В останні десятиліття ми стали свідками стрімкого розвитку різноманітних складних розподілених систем у фінансовій, промисловій, екологічній, безпековій, військовій та багатьох інших сферах застосування. Забезпечення високого рівня цілісності таких систем стає ключовим моментом їх розвитку, еволюції та використання, особливо в різних кризових та катастрофічних умовах, а також в умовах протистояння. У статті розглядається ряд існуючих робіт, присвячених цілісності, безпеці та відновленню розподілених систем. Також коротко описуються основні аспекти моделі та Технології просторового захоплення (ТПЗ) із відображенням загальних проблем парадигми, Мова просторового захоплення (МПЗ) та її мережева інтерпретація в розподілених середовищах. ТПЗ може динамічно встановлювати та підтримувати потужний контроль над великими розподіленими системами і, зокрема, створювати їх із нуля. Використовуючи представлення розподілених системних топологій на основі графів і вузлів, які мають як віртуальні, так і фізичні властивості, у статті представлено повне створення топології, починаючи з усіх вузлів паралельно, а потім з одного вузла, копіюючи існуючу топологію в подібних випадках. Також демонструється, як організувати розподілені системи таким чином, щоб вони могли самостійно відновлюватися за будь-яких обставин і після будь-яких пошкоджень, надаючи своїм вузлам універсальні генетичні можливості, за допомогою яких можна здійснювати будь-які самовідновлення. Таке відновлення може стосуватися як відсутніх сусідніх вузлів і зв'язків, так і всієї розподіленої топології, а це означає, що їх неможливо знищити навіть у найжорсткіших умовах. Ці особливості можуть бути особливо корисними у випадках пошкодження ІТ-мережі, екологічних і промислових катастроф, а також для врегулювання кризових ситуацій і становища на полях битв. У статті підтверджується ефективність розробленого підходу розподіленого керування для забезпечення високої цілісності та самовідновлення важливих розподілених систем.*
Ключові слова: *розподілені системи, цілісність системи, Технології просторового захоплення, сценарії, що самостійно розвиваються, самовідновлення розподіленої системи, вічність системи.*

Abstract. *In the last decades, we have witnessed an exploding growth of different kinds of sophisticated distributed systems with financial, industrial, ecological, security, military, and many other applications. Providing high integrity of such systems is becoming a key point of their development, evolution, and usage, especially in various crisis situations and under disastrous and adversarial conditions. The paper reviews a number of existing works on the integrity, security, and recovery of distributed systems. It also briefs the main aspects of the Spatial Grasp Model and Technology (SGT), reflecting some general issues of the paradigm, its Spatial Grasp Language (SGL), and networked SGL interpretation in distributed environments. SGT can dynamically establish and keep superior power over large distributed systems, including creating them from scratch. Using a graph-based representation of the distributed system topologies, with nodes having both virtual and physical properties, the paper shows full topology creation starting from all nodes in parallel and then from a single node, also copying the existing topology in similar cases. In addition, it demonstrates how to organize distributed systems in such a way so that they can self-recover in any circumstances and after any damages by supplying their nodes with universal genetic-like capabilities by which any self-repairs can be organized. Such recovery may be from missing neighboring nodes and links to the rebuilding of the distributed topologies, which means they cannot be destroyed even in the severest conditions. These features can be particularly useful after IT network damages, environmental and industrial disasters, for crisis management, and on battlefields. The paper confirms the efficiency of the developed distributed control approach for providing high integrity and self-recovery of important distributed systems.*

Keywords: *distributed systems, system integrity, Spatial Grasp Technology, self-evolving scenarios, distributed system self-recovery, system immortality.*

DOI: 10.34121/1028-9763-2023-2-18-27

1. Introduction

Especially in the last decades, we have witnessed an exploding growth of different kinds of sophisticated distributed systems with financial, industrial, ecological, security, military, and many other applications, covering both terrestrial and celestial spaces. Different application systems are also heavily integrated as worldwide super-systems that by their diversity, scale, and space coverage cannot be efficiently comprehended and managed from single points. Therefore, they need special robust distributed control, security, and survivability mechanisms. Providing high integrity of such systems is becoming a key point of their development, evolution, and usage, especially in various crisis situations and under disastrous and adversarial conditions. There are different meanings and interpretations of the word “integrity”. For example, it is considered a condition of a system where its mandated operational and technical parameters are within the prescribed limits, it performs the intended function in an unimpaired manner and implements protection mechanisms. From other definitions, the integrity of a system refers to the capability of performing correctly according to the original specification of the system, and so on.

The rest of the paper is organized as follows. Section 2 reviews a number of existing works on integrity, security, and recovery of distributed systems grouped as integrity and recovery in distributed systems; recovery, security, and self-recovery in distributed systems; self-repair in distributed systems; and self-healing in autonomous systems.

Section 3 briefs the main ideas of the Spatial Grasp Model and Technology (SGT), which reflect general technology issues, Spatial Grasp Language (SGL), and distributed SGL interpretation in distributed environments with any other details available in numerous existing publications on this paradigm.

Section 4 shows how to use SGL for the distributed system topology representation and creation, taking into account physical or virtual node coordinates, including full topology creation starting from all nodes in parallel, and full topology creation from a single node using a self-evolving spanning tree topology.

Section 5 demonstrates how to copy the already existing or newly created system topology and describes two cases: the first one starting from all nodes in parallel, and the second one – from a single node in a spanning-tree mode.

Section 6 describes how to create in SGL fully self-healing distributed topologies that can reconstruct themselves after simultaneous failures of any nodes if at least a single node remains alive. It provides a fully universal solution for system immortality which can be effectively used for numerous applications from IT to the fields of industry, security, and defense.

Section 7 concludes the paper by hinting at further research and implementation of the discussed ideas. References cite the analyzed publications on system integrity, as well as some books and the latest papers on SGT.

The aim of this paper is to investigate the applicability and efficiency of the Spatial Grasp Model and Technology developed and tested on numerous applications for organizing and supporting the integrity of large distributed systems, which may cover any physical and virtual spaces.

2. Existing approaches to the integrity, security, and recovery of distributed systems

The analyzed approaches, using mostly graph models for system representations and analysis, can be grouped by the following symbolic categories.

Integrity and recovery in distributed systems

Testing data integrity in distributed systems is described in [1]. Security threats in a distributed environment are one of the greatest threats in IT. The network provides the access path for both inside and outside attacks, which makes it the key access point for any type of security threat. Securing distributed infrastructure is not an easy task, it requires careful configuration and is subject to human errors. This paper presents a method of testing distributed environments against attacks on data integrity.

The integrity of distributed control systems is discussed in [2] which is focused on the integrity of the overall distributed control system. It classifies properties that enable verification and proof of the integrity of different subsystems. This classification is used to show how to protect the overall system integrity at different system levels. Based on an exemplary system in the domain of hydroelectric power plants, the paper shows practical examples of how to apply the results in the real world.

Data integrity and recovery management in cloud computing are analyzed in [3]. A new methodology is focused on and proposed for data recovery and data management to assure high-level scalability and reliability of fault recognition and fault tolerance. The offered methodology allows for segmenting data and generating tokens for the data split-up, where a missing segment of any faulty node can get back up from neighboring nodes.

Recovery, security, and self-recovery in distributed systems

Recovery and security in a distributed system are considered in [4]. Distributed systems have immense practical value in the computerized world and have many applications including scientific, engineering, commercial, and industrial ones. However, by their very nature of interconnectedness, distributed systems are subject to security problems and failures. These issues must be adequately addressed through effective techniques and methods to correct these problems.

A novel two-stage sequential disaster recovery strategy for resilient cyber-physical distribution power systems [5] is proposed with the consideration of cyber-physical collaborative optimization. In the first stage, a mixed-integer linear programming model is formulated based on the integration of a transportation network, a cyber network, and a physical network. In the second stage, resources are rescheduled to repair the remaining damaged components.

Self-recovery of a distributed system after a large disruption is analyzed in [6]. The work discusses the recovery of the systems hit by calamities, the self-repair of materials and tissues hit by radiation, and the ways how the structures have to be designed. A group of animals such as rabbits and squirrels was created where the rabbits and squirrels could take control over territory by grouping into clusters. Then a large disruption scared all the animals, and they started to scatter in all possible direction. The next stage was to analyze the recovery depending on agent features.

Self-repair in distributed systems

An approach for self-repair in a distributed system using immunity-based diagnostic mobile agents is offered in [7]. Self-repair has attracted much attention for fault tolerance in distributed computer network systems. In this paper, some units try to self-repair, that is, replace their data with data received from other units. Three different repair conditions are proposed. The simulations evaluate the effectiveness of such conditions in changing the numbers of initial abnormal hosts.

Minimum self-repairing graphs are discussed in [8]. A graph is self-repairing if it is two-connected and such that the removal of any single vertex results in no increase in distance between any pair of remaining vertices of the graph. The paper completely characterizes the class of minimum self-repairing graphs which have the fewest edges for a given number of vertices.

Self-healing in autonomous systems

A distributed formal-based model for self-healing behaviors in autonomous systems is discussed in [9]. Self-healing is one of the main features that characterize autonomic computing systems. Failure detection, recovery strategies, and reliability are of paramount importance to ensure continuous operation and correct functioning even in the presence of a given maximum amount of faulty components. A distributed formal model is offered for the specification, verification, and analysis of self-healing behaviors in autonomous systems.

Self-healing dilemmas in distributed systems with fault correction vs. fault tolerance are analyzed in [10]. Given the rise of distributed ledgers, edge computing, and the Internet of Things in several energy, transport, and health applications, measuring, understanding, and resolving self-healing dilemmas are timely challenges and critical requirements. The paper contributes a novel and general-purpose modeling of fault scenarios during system runtime. Methods are proposed to improve self-healing of large-scale decentralized systems at the design phase.

Self-healing networks with taking into account redundancy and structure are discussed in [11]. The paper introduces the concept of self-healing in the field of complex network modeling. In particular, self-healing capabilities are implemented through distributed communication protocols that exploit redundant links to recover the connectivity of the system. It analyzes the effect of the level of redundancy on the resilience to multiple failures, in particular, measures the fraction of nodes still served for increasing levels of network damages.

A decentralized self-healing approach for network topology maintenance is described in [12]. The paper proposes a multi-agent solution for recovering networks from node failures. To preserve the network topology, the proposed approach relies on local information about the network's structure which is collected and disseminated at runtime. These results validate the viability of the proposed self-healing solution, offering two variant implementations with diverse performance characteristics.

3. Spatial Grasp Model and Technology

General issues of the technology

Within the Spatial Grasp Model and Technology [13–26] a high-level operational scenario is represented as an active self-evolving pattern rather than a traditional program. This pattern is expressed in the recursive Spatial Grasp Language (SGL), it starts in any world point(s) (see Fig. 1) and propagates, replicates, modifies, covers, and matches the distributed environment in a parallel wave-like mode. This propagation also combines feedback echoing the reached control states and obtained data, which may be remote, for making decisions of higher levels, altogether providing holistic solutions unachievable by other models and systems. In Fig. 1, the parallel waves start from different world points Start 1 and Start 2 independently, then intersect (cooperating or competing) in space, whereas the waves from Start 3 develop from the area already covered from Start 2.

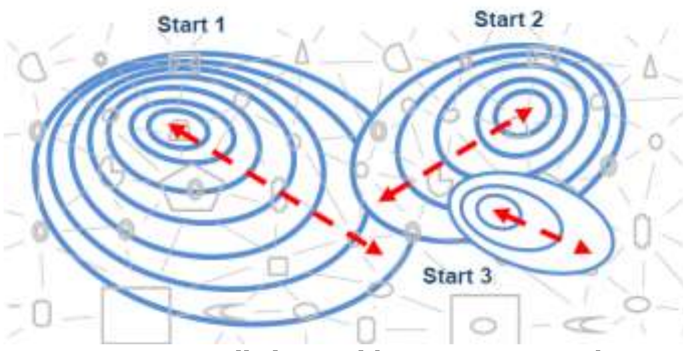

Figure 1 – Controlled wave-like space coverage by an active recursive SGL code

Spatial Grasp Language (SGL)

The SGL allows for expressing direct space presence and operations with unlimited parallelism. Its universal recursive organization with operational scenarios called *grasp* can be expressed just by a single string:

grasp $\rightarrow$ *constant* | *variable* | *rule* ({ *grasp*, })

The SGL rule expresses certain actions, control, descriptions, or contexts accompanied by operands that can themselves be any *grasp* too. Top SGL may be summarized as follows, with details of the language usage found in many existing publications, including [13–26].

constant $\rightarrow$ *information* | *matter* | *custom* | *special*
variable $\rightarrow$ *global* | *heritable* | *frontal* | *nodal* | *environmental*
rule $\rightarrow$ *type* | *usage* | *movement* | *creation* | *echoing* |
verification | *assignment* | *advancement* | *branching* |
transference | *exchange* | *timing* | *qualifying*

Distributed SGL interpretation

Each SGL interpreter copy can handle and process multiple active SGL scenario codes that freely evolve and propagate in space and between the interpreters. Integrated with any distributed system, the SGL interpretation network can form a spatial computer with unlimited power for the simulation and management of distributed systems and worlds. Such collective engines can simultaneously execute many cooperative and competitive tasks without any central resources or control.

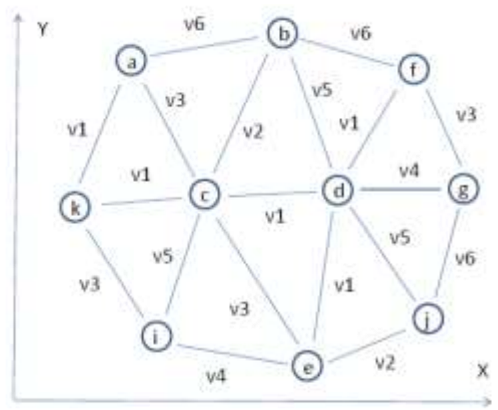

Figure 2 – Representation of the topology of distributed systems in a physical-virtual graph mode

4. Distributed system topology representation and creation

A topology example in the form of a graph (which is distributed in two-dimensional physical space) with the named nodes and links is shown in Fig. 2.

Representing topology with node coordinates

A compact textual representation of this topology assigned to variables `Top` (with node names followed by lists of named links leading to named neighbors) and `Loc` (having node names *w* followed by their *Cw* location coordinates expressed in some physical or virtual notation) can be 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)
```

Creating a full topology starting from all nodes in parallel

Creating only single nodes (without connections with neighbors) in proper space locations (with known coordinates) can be done in parallel by the following SGL text where operation `split` creates as many parallel branches as there are elements in the embraced variable `Loc`, with each element addressed by SGL environmental variable `VAL` (short for `VALUE`) in the corresponding branch:

```
split(Loc); create_node(VAL[1], coord(VAL[2]))
```

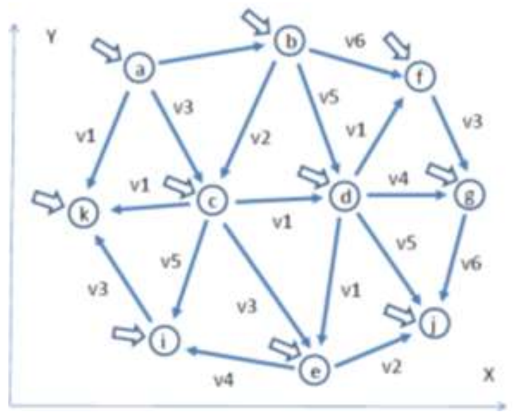

Figure 3 – Creating a distributed networked structure in parallel

Creating a full topology in parallel (i.e. all nodes with the named links to other nodes, as well as nodes having names and coordinates in virtual or physical space) can be done by the following scenario (resolving competition of neighboring nodes attempting to create the same link between them) (see also Fig. 3):

```
Top = …; frontal(Loc) = …;
align(split(Top); frontal(NN) =
VAL[2];
      create_node(VAL[1],
coord(Loc:VAL[1]));
split(NN); NAME > VAL[2];
linkup(VAL[1], node(VAL[2],
coord(Loc:VAL[2])))
```

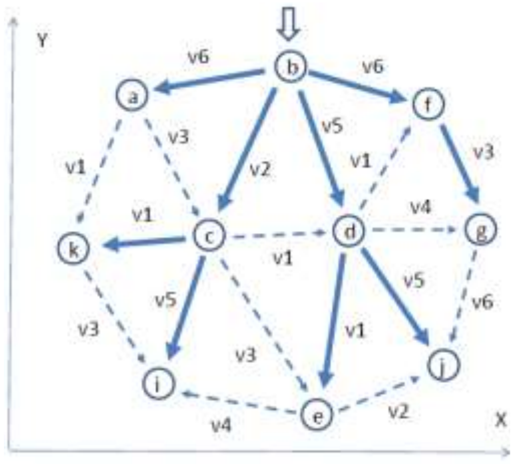

Figure 4 – Creating a distributed networked structure from a single node in a spanning-tree mode

Creating a full topology starting from a single node

Creating the same topology but starting from a given node and using a spatially evolving spanning-tree mode may be done by the following scenario (see also Fig. 4):

```
Top = …; frontal(Loc) = …; Start = …;
create_node(Start, coord(Loc:Start));
repeat(
  split(Top:NAME);
  or_seq((empty(see(Loc:VAL[2]));
          create(link(VAL[1],
node(VAL[2], coord(Loc:VAL[2])))),
         (NAME > VAL[2];
          linkup(VAL[1], node(VAL[2],
coord(Loc:VAL[2]))); quit)))
```

5. Copying a system topology

Copying a full topology starting from all nodes in parallel

Copying the already existing or just created full topology starting from all its nodes in parallel and obtaining its textual representation in the already mentioned variables `Top` and `Loc` (with the search `Area` defined by 3D coordinates or any other descriptions and zone addresses used in virtual spaces) can be done as follows:

```
Area = (X1_X2, Y1_Y2, Z1_Z2);
Top = (hop_nodes(Area, all);
       append(NAME, unit(hop_links(all); LINK & NAME)));
Loc = (hop_nodes(Area, all); NAME & WHERE)
```

Fig. 3 (except for the links creating directions, which may differ) can also be used to demonstrate the work of this scenario.

Copying a full topology starting from a single node

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

```
Area = …; hopfirst_node(Area, any);
Top = repeat(blind(append(NAME, unit(hop_links(all); LINK & NAME))),
             hopfirst_links(all));
hopfirst_node(Area, any);
Loc = repeat(blind(NAME & WHERE), hopfirst_links(all))
```

To demonstrate this scenario, we can also use Fig. 4 (ignoring, however, the links creating directions, which may differ).

6. Fully self-healing topologies

We can also easily write in SGL a full recovery scenario which, initially applied in all nodes, will allow the whole topology to self-recover after arbitrary damages (if even a single node remains alive). Seeing the absence of its neighbors and/or links to them, each node will provide their restoration, including supplying the restored nodes with the ability to do the same for the absent neighbors if such are noticed too, and so on. This makes the whole topology alive and self-healing, regardless of damages that may appear at any time and simultaneously in many nodes. Fig. 5 and the following SGL scenario explain these unique self-healing capabilities, actually allowing for the "immortality" of distributed system topologies under SGL and SGT.

```
frontal(Top = …, Loc = …, Business = {…},
 Recovery =
  {split(Top:NAME);
   repeat_stay(
     sleep(delay);
     empty(see(Loc:VAL[2]));
     create(link(VAL[1], node(VAL[2], coord(Loc:VAL[2]))));
     free(run(Business, Recovery)))});

Area = …; hop_nodes(Area, all); run(Business, Recovery)
```

The scenario given above uses the universal recovering facility as a procedure assigned to the variable `Recovery` that together with another (traditional) node business procedure in the variable `Business` is always transferred to the just recovered nodes and activated there (both procedures are spatially transferable in `frontal` variables). For such a scenario, any number of nodes can be killed at any time and even simultaneously like, for example, the following eight nodes:

```
Area = …; delete(hop_nodes(Area, (a, b, c, d, g, i, e, j)))
```

The full self-recovery of the topology after such damage will simultaneously originate in two remaining nodes `k` and `f` as shown in Fig. 6, spreading afterward to all other dynamically restored nodes that, if needed, in their turn, spread this further, and so on.

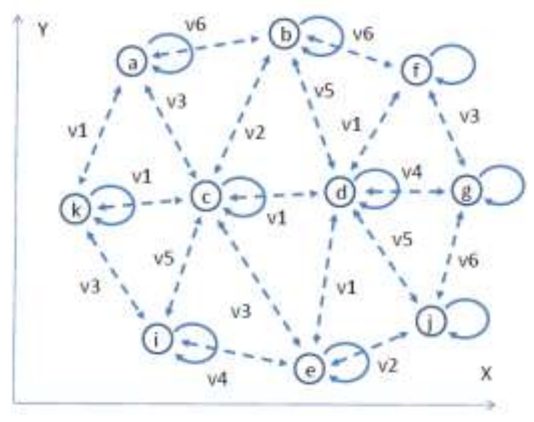

Figure 5 – Making distributed networked systems constantly self-analyzing and self-repairing after any damages

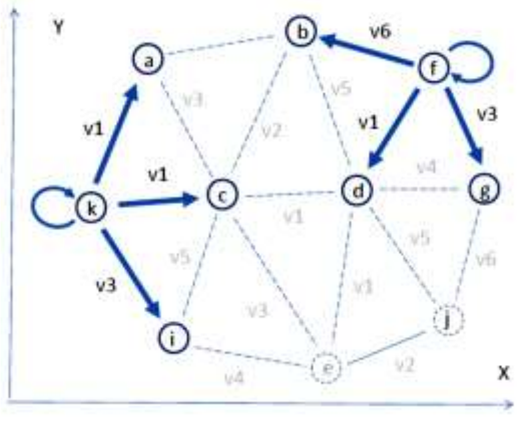

Figure 6 – Full self-recovery of a distributed system starting from two remaining nodes

7. Conclusions

The paper has investigated the application of the distributed Spatial Grasp Model and Technology for the management of large distributed networked systems from their expression, creation, copying, and modeling to the continuous self-analysis and self-recovery after any damages. Based on the compact recursive self-spreading, self-matching, and self-evolving operational code, SGT can provide the highest possible integrity of distributed dynamic systems unachievable by any other models and technologies, which are usually based on communicating parts or agents, with known difficulties of achieving the needed parameters and behavior. The super-virus mode of operation of the offered approach, where any holes and damages can be immediately self-repaired and covered by the recursive self-spreading active code, can make the large distributed dynamic systems actually "immortal" in many important applications. The latter may cover the fields of industry, finance, security, environment, defense, space conquest, etc. Taking into account our experience with its previous versions in different countries, we can say that the offered distributed control technology can be easily implemented on any existing platforms and integrated with other networked systems, forming altogether powerful spatial engines effectively operating in both terrestrial and celestial environments.

Acknowledgments

I am especially thankful to my good colleagues, advisers, and friends from different countries, who supported the ideas expressed in this paper as well as other publications on the Spatial Grasp Technology in scientific journals and books: Gabriella Williams (UK), Thomas Ditzinger (Germany), John Page (Australia), Steve Lambacher (Japan), Kohei Arai (Japan), Bob Nugent (US), Son Vuong (US, Vietnam), and James White (US, Hawaii).

REFERENCES

1. Mittal M., Sangani R., Srivastava K. Testing Data Integrity in Distributed Systems. *Procedia Computer Science*. 2015. Vol. 45. P. 446–452. URL: https://www.sciencedirect.com/science/article/pii/S1877050915003130.

2. Rauter T. Integrity of Distributed Control Systems. *Student Forum of the 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. Toulouse, France, 2016. June. URL: https://hal.science/hal-01318372/file/DSN-Student-Forum_%237_Integrity-of-Distributed-Control-Systems.pdf.
3. Sivanandham G., Gnanasekar J.M. Data Integrity and Recovery Management in Cloud Systems. *Proc. of the Fourth International Conference on Inventive Systems and Control (ICISC 2020)*. URL: https://www.researchgate.net/publication/343751352_Data_Integrity_and_Recovery_Management_in_Cloud_Systems.
4. Sodhi G.K. Recovery and Security in Distributed System. *International Journal of Advanced Research in Computer and Communication Engineering*. 2015. Vol. 4, Issue 12. URL: https://www.ijarcce.com/upload/2015/december-15/IJARCCE%20105.pdf.
5. Sun X., Chen J., Zhao H., Zhang W., Zhang Y. Sequential Disaster Recovery Strategy for Resilient Distribution Network Based on Cyber-Physical Collaborative Optimization. *IEEE Transactions on Smart Grid*. 2023. Vol. 14, N 2. P. 1173–1187. DOI: 10.1109/TSG.2022.3198696. URL: https://ieeexplore.ieee.org/document/9857641.
6. Popa-Simil V., Poston H., Thomas Mrs., Popa-Simil L. Self-recovery of a distributed system after a large disruption. New Mexico Supercomputing Challenge: Final Report. 2012. April 1. URL: http://www.supercomputingchallenge.org/11-12/finalreports/15.pdf.
7. Watanabe Y., Sato S., Ishida Y. An Approach for Self-Repair in Distributed System Using Immunity-Based Diagnostic Mobile Agents. *Knowledge-Based Intelligent Information and Engineering Systems:* 8th International Conference, KES 2004. Wellington, New Zealand, 2004. September 20–25. URL: https://link.springer.com/chapter/10.1007/978-3-540-30133-2_66.
8. Farley A.M., Proskurowski A. Minimum Self-Repairing Graphs. *Graphs and Combinatorics*. 1997. Issue 13. P. 345–351. DOI: https://doi.org/10.1007/BF03353012. URL: https://link.springer.com/article/10.1007/BF03353012.
9. Hafaiedh I.B., Slimane M.B. A distributed formal-based model for self-healing behaviors in autonomous systems: from failure detection to self-recovery. *The Journal of Supercomputing volume.* 2022. Vol. 78. P. 18725–18753. URL: https://link.springer.com/article/10.1007/ s11227-022-04614-0.
10. Nikolic J., Jubatyrov N., Pournaras E. Self-Healing Dilemmas in Distributed Systems: Fault Correction vs. Fault Tolerance. *Journal of Latex Class Files*. 2021. Vol. X, N X. URL: https://arxiv.org/pdf/2007.05261.pdf.
11. Quattrociocchi W., Caldarelli G., Scala A. Self-Healing Networks: Redundancy and Structure. *PLOS One*. 2014. Vol. 9, Issue 2. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0087986.
12. Rodriguez A., Gomez J., Diaconescu A. A decentralized self-healing approach for network topology maintenance. *Autonomous Agents and Multi-Agent Systems*. 2021. Vol. 35. P. 1743–1745. URL: https://link.springer.com/article/10.1007/s10458-020-09486-3.
13. Sapaty P.S. A distributed processing system: European Patent N 0389655. Publ. 10.11.93. European Patent Office. 35 p.
14. Sapaty P.S. Mobile Processing in Distributed and Open Environments. New York: John Wiley & Sons, 1999. 436 p.
15. Sapaty P.S. Ruling Distributed Dynamic Worlds. New York: John Wiley & Sons, 2005. 255 p.
16. Sapaty P.S. Managing Distributed Dynamic Systems with Spatial Grasp Technology. Springer, 2017. 284 p.
17. Sapaty P.S. Holistic Analysis and Management of Distributed Social Systems. Springer, 2018. 234 p.
18. Sapaty P.S. Complexity in International Security: A Holistic Spatial Approach. Emerald Publishing, 2019. 160 p.
19. Sapaty P.S. Symbiosis of Real and Simulated Worlds under Spatial Grasp Technology. Springer, 2021. 251 p.
20. Sapaty P.S. Spatial Grasp as a Model for Space-Based Control and Management Systems. CRC Press, 2022. 280 p.
21. Sapaty P.S. The Spatial Grasp Model: Applications and Investigations of Distributed Dynamic Worlds. Emerald Publishing, 2023. 184 p.

22. Sapaty P.S. Distributed Control Technology for Air and Missile Defence Operations. *International Relations and Diplomacy*. 2022. Vol. 10, N 4. P. 141–157. DOI: 10.17265/2328-2134/2022.04.001. URL: https://www.davidpublisher.com/Public/uploads/Contribute/ 6364741a7b432.pdf.
23. Sapaty P.S. Relation of Spatial Grasp Paradigm to Higher Psychological and Mental Concepts. *Acta Scientific Computer Sciences*. 2022. Vol. 4, Issue 12. URL: https://actascientific.com/ASCS/pdf/ASCS-04-0359.pdf.
24. Sapaty P.S. Seeing and Managing Distributed Worlds with Spatial Grasp Paradigm. *Acta Scientific Computer Sciences*. 2022. Vol. 4, Issue 12. URL: https://actascientific.com/ ASCS/pdf/ASCS-04-0365.pdf.
25. Sapaty P.S. Comprehending Distributed Worlds with the Spatial Grasp Paradigm. *Mathematical machines and systems*. 2022. N 1. P. 12–30. URL: http://www.immsp.kiev.ua/ publications/articles/ 2022/2022_1/01_22_Sapaty.pdf.
26. Sapaty P.S. Spatial management of air and missile defence operations. *Mathematical machines and systems.* 2023. N 1. P. 30–49. URL: http://www.immsp.kiev.ua/publications/articles/2023/2023_ 1/01_23_Sapaty.pdf.

Стаття надійшла до редакції 26.03.2023