International Relations and Diplomacy, May-June 2025, Vol. 13, No. 3, 164-179
doi: 10.17265/2328-2134/2025.03.004

Spatial Grasp Model for Distributed Management and Its Comparison With Traditional Algorithms

Peter Simon Sapaty
National Academy of Sciences of Ukraine, Kyiv, Ukraine

The word “spatial” fundamentally relates to human existence, evolution, and activity in terrestrial and even celestial spaces. After reviewing the spatial features of many areas, the paper describes basics of high level model and technology called Spatial Grasp for dealing with large distributed systems, which can provide spatial vision, awareness, management, control, and even consciousness. The technology description includes its key Spatial Grasp Language (SGL), self-evolution of recursive SGL scenarios, and implementation of SGL interpreter converting distributed networked systems into powerful spatial engines. Examples of typical spatial scenarios in SGL include finding shortest path tree and shortest path between network nodes, collecting proper information throughout the whole world, elimination of multiple targets by intelligent teams of chasers, and withstanding cyber attacks in distributed networked systems. Also this paper compares Spatial Grasp model with traditional algorithms, confirming universality of the former for any spatial systems, while the latter just tools for concrete applications.

Keywords: spatial awareness, spatial control, spatial consciousness, Spatial Grasp Technology, Spatial Grasp Language, spatial scenarios, cyber attacks, distributed algorithms, mobile agents

1. Introduction

The humankind exists, operates, and develops in large distributed spaces, where “spatial” may be considered as the basic and dominant feature philosophically, conceptually, and practically. To prosper in this spatial continuum, adequate high level methodologies, organizations, and tools must be developed and used.

The aim of this paper is to describe a universal high-level model and technology for dealing with spatial systems of different natures and show advantages of the offered spatial paradigm over traditional algorithms.

The rest of the paper is organized as follows:

Section 2 reviews “spatial” as the basic feature of human activity including spatial temporal analysis, space-time continuum, spatial analysis, spatial vision, spatial feeling, spatial understanding, spatial awareness, spatial consciousness, spatial ability, spatial intelligence, spatial knowledge, spatial danger, spatial warfare, spatial management, spatial planning, spatial belief, spatial faith, spatial hate, spatial crime, spatial diversity, spatial hope, and spatial virus. Section 3 describes basics of the universal model and technology oriented from the very beginning on seeing, staying, moving, and making operations in spaces of any complexity, integrity, and coverage. It explains the main technology ideas, its basic Spatial Grasp Language (SGL), how SGL scenarios self-evolve in physical, virtual, or imaginable spaces, and main details of the networked technology implementation allowing

---

Peter Simon Sapaty, Ph.D., Chief Research Scientist, Institute of Mathematical Machines and Systems, National Academy of Sciences of Ukraine, Kyiv, Ukraine.

the whole world to be converted into a powerful spatial engine. Section 4 describes examples of typical solutions in SGL for distributed dynamic systems which include: Finding shortest path tree (SPT) from a node to all other nodes, also shortest path between two nodes, finding and collecting information on certain individuals worldwide identified by specific features, and how the team of chasers is fighting multiple targets under dynamically supported global awareness in the team. Section 5 reviews existing cybersecurity publications and offers practical scenario examples in SGL for withstanding cyber attacks, including discovering sources of viruses in distributed networks. Section 6 compares the Spatial Grasp model with traditional algorithms usually considered as a sequence of instructions, procedure for solving a problem, set of rules to be followed, step-by-step procedure, etc. It also mentions some existing algorithmic extensions such as distributed algorithms, mobile agents, and spatial analysis algorithms. Section 7 concludes the paper. References contain numerous sources on which the paper is based, and Appendix provides the Spatial Grasp Language summary.

2. "Spatial" as the Basic Feature of Human Activity

We are starting here with *spatial temporal analysis* describing the ontological status of something that exists in both space and time (Oliver, 2019). It also relates to *space-time continuum* that fuses the three dimensions of space and one of time (Wikipedia, 2025a). *Spatial analysis* relates to knowledge, skills, and habits of mind to use concepts of space (such as distance, orientation, etc.); used as an umbrella term covering spatial perception, spatial ability, visual perception, or spatial intelligence (IGI, 2025; Geometriedidaktik; 2025). *Spatial vision* refers to the ability of an eye to capture and process fine details of a visual scene (ScienceDirect, 2025). *Spatial feeling* uses rich illustrations and examinations of art, technology, and philosophy to explain this phenomenon (Bardt, 2024). *Spatial understanding* contains historical background and a discussion of space missions, space environment, orbits, etc., also provides fundamentals of space missions and systems, a complete picture of the space industry (Sellers, Astore, & Giffen, 2000; 2015). *Spatial awareness* is the ability to be aware of your surroundings and where you are in relation to the surrounding objects, as well as the ability to understand your body's position in relation to your surroundings (Twinkl, 2025; Seladi-Schulman, 2020). *Spatial consciousness* may refer to individual or collective awareness about real-world spatial phenomena and processes; it relates to the mental process that allows us to perceive, organize, and navigate the space around us (Galland & Grønning, 2019; Karmarkar, 2023). *Spatial ability* is the capacity to understand, reason, and remember the visual and spatial relations among objects or space (Wikipedia, 2025c). *Spatial intelligence* is the concept of being able to successfully perceive and derive insight from visual data (Fitzgibbons, 2019). *Spatial knowledge* refers to chunks of knowledge that are focused on the spatial component like coordinates and place names (Laurini & Thompson, 1992). *Spatial danger* originates from the risk that an infection in this space can spread easily to either side (Wikipedia, 2025d). *Spatial warfare* includes ground-to-space warfare, space-to-space warfare, and space-to-ground warfare (Wikipedia, 2025e). *Spatial management* highlights the need for more interdisciplinary, strategic, and collaborative methods to achieve broad goals (Dandoulaki & Lazoglou, 2023). *Spatial planning* mediates between the respective claims on space of the state, market, and community (Wikipedia, 2025f). *Spatial belief* is a central and researched social category, as an educational category in the school context; it also extends the cognitive research on human belief to the area of spatial reasoning (University of Passau, 2024; Nejasmic, Bucher, & Knauff, 2015). *Spatial faith* has a long history from its roots in the ancient drafting of religious cosmologies; also, astronauts observed their religions while in space, sometimes publicly, sometimes privately (Corrigan, 2008; Wikipedia, 2025g). *Spatial hate* is examined in different perspectives and variety of spatial and temporal contexts.

Examining hate crime from a spatial and temporal perspective allows us to understand it better (Hopkins, 2022; Wenger & Lantz, 2022). *Spatial crime* explains that many crime incidents may happen at particular spatial locations and points in time (Hipp, 2022). *Spatial diversity* is widely used in mobile radio base stations, taking advantage of random nature of waves propagation (ScienceDirect, 1998). *Spatial hope* may center on geostationary orbit, its management, and challenges associated with its scarcity (Roy, 2020). *Spatial virus* is based on exploitation of multiple scales in space and time, whereas many investigations describing dynamics of infectious are based on spatially structured models (Kumberger, Frey, Schwarz, & Graw, 2016; Funk, Jansen, Bonhoeffer, & Killingback, 2005).

3. Spatial Grasp Model and Technology

Key Ideas

Within Spatial Grasp Technology (SGT) (see the related patent (Sapaty, 1993), published books (Sapaty, 1999; 2005; 2017; 2018; 2019; 2021; 2022; 2023; 2024a; 2024b; 2025), early history (Bondarenko, Mikhalevich, Nikitin, & Sapaty, 1970; Sapaty, 1973; 1986; 1996)), a high-level operational scenario in recursive Spatial Grasp Language (SGL), starting in any world points, propagates, covers, and matches the distributed environment in parallel wavelike mode, as symbolically shown in Figure 1. Such propagation can result in returning and analyzing the reached states and data as well as in further waves from the initial and new nodes, which may be arbitrarily remote and in any numbers.

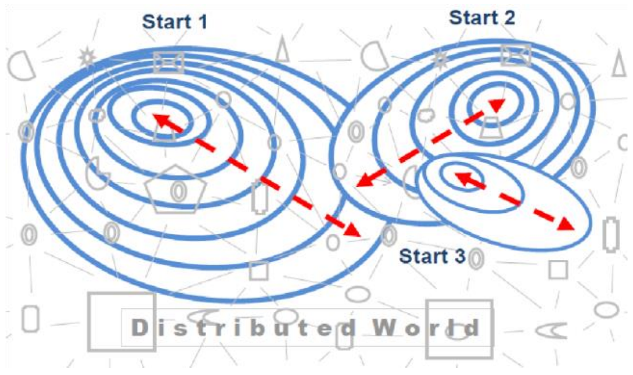

Figure 1. Symbolic views of wavelike world coverage under SGT.

This concept is based on quite different philosophy and practice of traditional dealing with large distributed and parallel systems. Instead of representing systems and solutions in them in the form of communicating parts or agents the developed Spatial Grasp paradigm is organizing everything by *integral, holistic, and parallel substance self-propagating thru and covering-matching distributed worlds* of different natures. The latter may include: p*hysical world*, *virtual world*, *executive world*, also their combinations.

Spatial Grasp Language

The recursive language top level organization can be expressed just in a single string-formula mode:

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

which can be further extended as follows:

constant	$\rightarrow$	*information* | *matter* | *special*
variable	$\rightarrow$	*global* | *heritable* | *frontal* | *nodal* | *environmental*
rule	$\rightarrow$	*movement* | *creation* | *advancement* | *branching* | *cycling* | *echoing* | *verification* |

assignment | *transference* | *exchange* | *timing* | *qualifying* | *type* | *usage*

Some details on its components are as follows:

Constant can be self-identifiable by the way written or defined by special rules embracing them with arbitrary textual representations, being of *information*, *matter,* and *special*.

Variable types can be defined by how their names are written, but in general, variables may have any names if their types are declared by special rules, being of *global, heritable*, *frontal*, *nodal,* and *environmental*.

Rule may be further detailed as follows. *Movement* may result in virtual hopping to the existing nodes (the ones having virtual or/and executive dimensions) or in real movement to new physical locations, subsequently starting the remaining scenario in the nodes reached. *Creation* creates or removes nodes and/or links leading to them during distributed world navigation; after termination, the resultant values correspond to the names of reached nodes, and the next scenario steps may start from all these nodes. *Advancement* can organize forward advancement in space and time of the embraced scenarios; they can evolve within their sequence in synchronous or asynchronous manner. *Branching* rules allow the embraced set of scenario operands to develop "in breadth", each from the same starting position, with the resultant set of positions and order of their appearance depending on the logic of a concrete branching rule. *Cycling* repeatedly invokes the embraced scenario; the resultant set of positions on the rule will be integration of all positions from successful scenario invocations from the same point, or those obtained on its last invocation in a sequence. *Echoing* oriented on various aspects of data and knowledge processing containing rules which may use local or remote values for different operations to be processed in the starting points. *Verification* rules verify the result of concrete procedure while remaining after their completion in the same world positions where they started. *Assignment* rules assign the result of the right scenario operand, which may be arbitrarily remote, to the variable or set of variables named or reached by the left scenario operand, which may be remote too. *Transference* rules organize transference of control in distributed scenarios to the code treated either as SGL procedure or an external system; they can also trigger broadcasting of the data obtained to different destinations. *Exchange* rules provide input of external information or physical matter by the initiative of SGL scenario, also output of the resultant value obtained by the embraced scenario outside or to other nodes. *Timing* rules are dealing with conditions related to a time allowed for the scenarios they embrace, or provide the needed time delays between different scenario parts. *Qualifying* rules are providing or supporting certain qualities or abilities, also setting constraints or restrictions to the scenarios they embrace. *Type* rules explicitly assign types to different constructs, using their existing repertoire. *Usage* rules explain how to use the information units they embrace.

For more SGL details please see the Appendix.

Self-evolving SGL Scenarios

The following are some hints on how SGL scenarios self-evolve in distributed environments. They are developing in *steps, potentially parallel,* with new steps produced on the results of previous steps. Any step is associated with a *certain point or points* of the world. Each step provides a resultant *value* (single or multiple) and resultant *control state*. Different scenario parts may evolve from the same points in *ordered, unordered, or parallel* manner, as independent or interdependent branches. Different scenario parts can spatially *succeed each other,* with new parts evolving from positions reached by the previous parts. This potentially parallel and distributed scenario can proceed in *synchronous or asynchronous* modes, also any combinations. SGL operations and decisions can use *control states* and *values* returned from subsequent scenario parts, effectively combining controlled forward and backward scenario evolution. Different steps from the same or different scenarios may happen to be temporarily associated with the *same world points*, while sharing persistent or provisional information in them. Staying with different world points, the scenarios can *analyze and impact* the navigated worlds via these locations. Scenarios navigating distributed spaces can form active *distributed infrastructures* in them, which can be shared with other scenarios. Overall organization of the world creation, coverage, modification, analysis, and processing provided by SGL rules can be *arbitrarily nested*. The evolving SGL scenarios can *abandon* utilized parts if not needed any more, also self-modify and self-replicate during space navigation. The special control states (including *thru*, *done*, *fail*, *fatal*) appear after completion of different scenario steps, indicating their progress or failure. They can be used for effective control of multiple distributed processes with proper decisions at different levels.

Networked SGL Implementation

Each SGL interpreter copy can handle and process multiple active scenario code propagating in space and time. Communicating interpreters can be of arbitrary number of copies (up to thousands and millions if needed) effectively integrated with other existing systems and communications, altogether representing powerful spatial engines operating without central resources or control. Hardware or software SGL interpreters, shown in Figure 2 as universal control and processing units (working with complex spatial graph and network data) can be installed, runtime created too, in proper physical or virtual world positions.

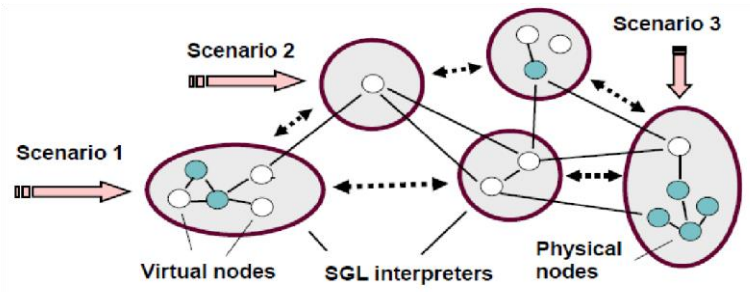

Figure 2. Distributed SGL interpretation working with complex spatial data.

The SGL interpreter's main components and its general organization are shown in Figure 3.

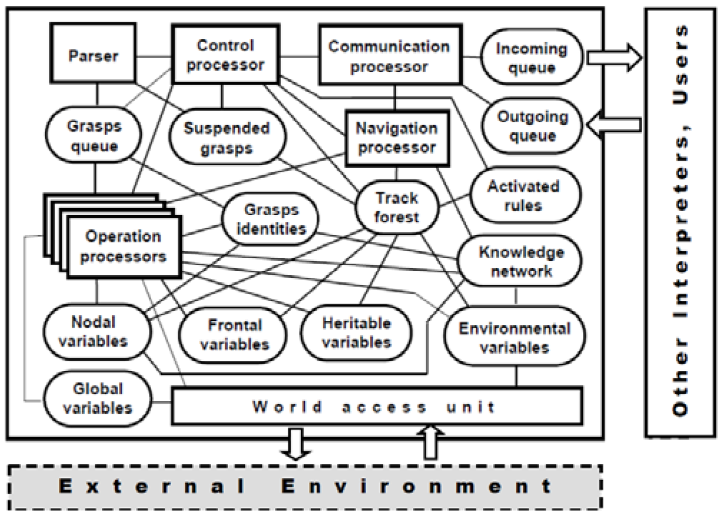

Figure 3. SGL interpreter's main components and their interactions.

The interpreter consists of a number of specialized *functional processors* (shown by rectangles) working with and sharing specific data structures. These include: Communication Processor (CP), Control Processor (COP), Navigation Processor (NP), Parser (P), different Operation Processors (OP), and special (external & internal) World Access Unit (WAU) directly manageable from SGL. Main *data structures* (also referred to as *stores*) with which these processors operate (shown by ovals) comprise: Grasps Queue (GQ), Suspended Grasps (SG), Track Forest (TF), Activated Rules (AR), Knowledge Network (NN), Grasps Identities (GI), Heritable Variables (HV), Fontal Variables (FV), Nodal Variables (NV), Environmental Variables (EV), Global Variables (GV), Incoming Queue (IQ), and Outgoing Queue (OQ).

As both *backbone and nerve system* of the distributed interpreter, its self-optimizing *Spatial Track System* provides hierarchical command and control as well as remote data and code access. It also supports spatial variables and merges distributed control states for decisions at higher organizational levels. The track infrastructure is automatically distributed between active components (humans, robots, computers, smart-phones, satellites, etc.) during scenario self-spreading in distributed environments. It integrates the following operational stages: *forward grasping*, *echoing*, and *further forward development.*

4. Examples of Typical Solutions in SGL

Finding Shortest Path Tree (SPT) From a Node to All Other Nodes

Imagine we have a distributed physical or virtual network, as of Figure 4 (which can be effectively built in SGL).

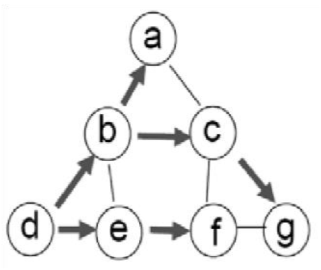

Figure 4. A network with shortest path tree on it.

To create on this network a tree structure starting from some node "d" and providing shortest paths from it to all other nodes (as in Figure 4, too) we may write in SGL the following scenario (the found SPT will be embedded into the network structure with variables "Up" in all nodes):

```
nodal(Dist, Up); hop_node(d); Distance = 0; frontal(Far);
repeat(
   hop_links(all); Far += 1;
   or(Distance == nil, Distance > Far);
   Distance = Far; Up = BACK)
```

This scenario is based on *virus-like parallel propagation* of the self-modified SGL code starting in certain node like "d" and then self-covering-conquering the whole network structure. After its completion, we may start in any node, say "c", and follow contents of nodal variables "Up" leading to the above nodes of the obtained tree, finally receiving the shortest path to this node from the starting node "d" as follows:

```
hop_node(c); frontal(Spath) = NAME;
```

```
repeat(hop(Up); append(NAME, Spath); output(Spath)
```

The output result will be (d, b, c).

Finding Certain Individuals Worldwide

Imagine we have to find detailed information about individuals belonging to some "Group" identified by specific "features", and originating in "START" position represented by physical or virtual address. Staying in it, the group members can be found by their "features" in "local_databases". This may fail to find records on some or all individuals sought, but their "traces" may exist in "local_security" systems. If such "traces" exist and lead to known "Other" world locations, we may search both data and security records at other points too, and so on, with the checking potentially spreading and covering the whole world. The found "match" from different points can be collected and returned (with whereabouts of individuals) to the "START" point with final "output" there. This scenario can be expressed in SGL as follows (with its possible spatial coverage shown in Figure 5 where universal SGT interpreters U are supposed existing in all nodes).

```
hopfirst(START); nodal(Other);
frontal(Group) = features;
output('Records found worldwide:' &&
   repeat(free(match(Group, local_databases)),
   Other = traces(Group, local_security);
   hopfirst(Other)))
```

This scenario first employs *forward parallel virus-like world propagation* as in previous example, and then in a *feedback propagation* brings to the starting node all distributed results, using for this the spatial track system feature of the networked SGT interpreter.

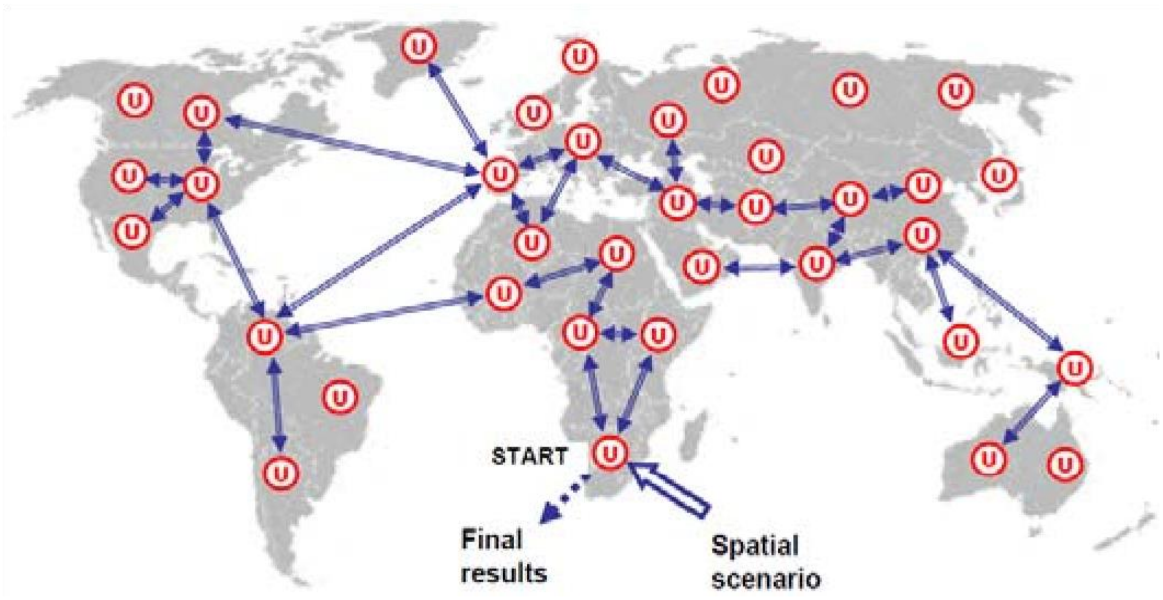

Figure 5. Spatial worldwide search and collection for certain individuals.

Answer in the START point may be as follows:

Records found worldwide: *match_1*, *match_2*, …, *match_m*

Chasers Fighting Targets Under Global Situational Awareness

By *regularly enriching* the distributed swarm of chasers with a sort of global awareness over the operational area we may essentially improve its performance. This awareness can be effectively embedded into communicating chasers where the *targets seen* by individual chasers can be regularly exchanged with their neighbors, as in Figure 6.

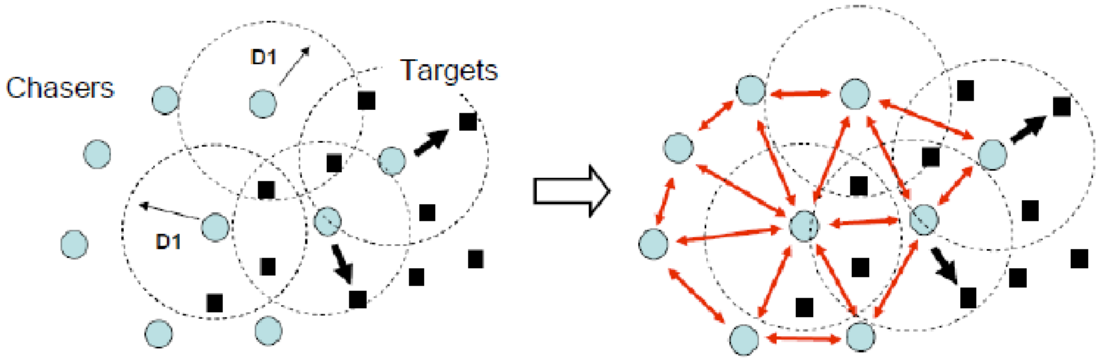

Figure 6. Fighting targets under growing global awareness.

This makes all “chasers” *gradually aware* of all “targets” in the region despite not all directly visible individually (“D1” as vision threshold), always organizing their movement *to be closer* to the targets, and then “select,_move”, and “destroy” them. An example of such SGL solution may be as follows:

```
hop_chasers(all);
nodal(D1 = distance, Targets); frontal(Exchange);
repeat(
  extend(Targets) = search(D1);
  select_move_destroy_remove(Targets);
  stay(Exchange = Targets; hop(neighbors);
    merge(Targets, Exchange));
  sleep(Delay))
```

More details and explanations on these and many other applications of the SGT and programming in SGL can be found in many existing technology related publications, including (Sapaty, 1999; 2005; 2017; 2018; 2019; 2021; 2022; 2023; 2024a; 2024b; 2025).

5. Cybersecurity Applications of SGT

Cybersecurity and Cyberattacks

Cybersecurity (Cisco, 2025) is the practice of protecting systems, networks, and programs from digital attacks. Implementing effective cybersecurity measures is particularly challenging today because there are more devices than people, and attackers are becoming more innovative. Common internet cyber attacks (Shutterstock, 2025) may be symbolically depicted as in Figure 7.

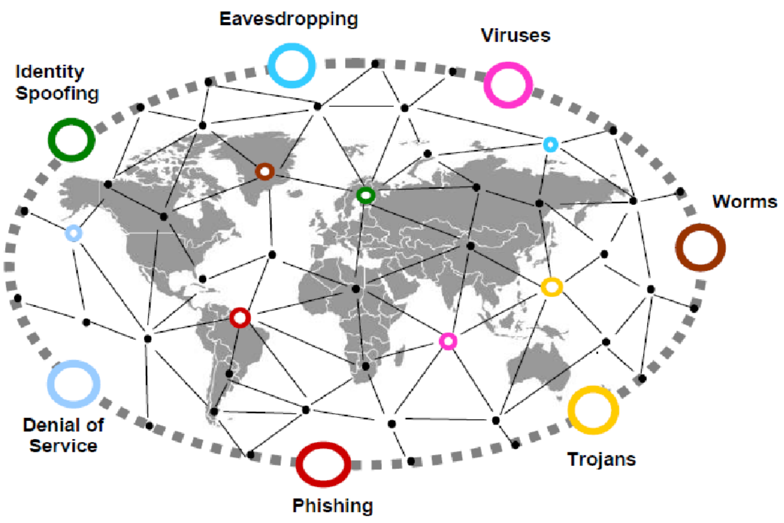

Figure 7. Typical internet cyber attacks.

A *cyber attack* usually refers to an action designed to target a computer or any element of a computerized information system to change, destroy, or steal data, as well as exploit or harm a network. Cyber attacks have been on the rise, in sync with the digitization of business that has become more and more popular in recent years. More types of cyber attacks can be found in Fortinet (2025) with their short review following.

DoS and DDoS attacks. A denial-of-service (DoS) attack is designed to overwhelm the resources of a system to the point where it is unable to reply to legitimate service requests.

MITM attacks. Man-in-the-middle (MITM) types of cyber attacks refer to breaches in cybersecurity that make it.

Phishing attacks. A phishing attack occurs when a malicious actor sends emails that seem to be coming from trusted, legitimate sources in an attempt to grab sensitive information from the target.

Whale-phishing attacks. A whale-phishing attack is so-named because it goes after the “big fish” or whales of an organization, which typically include those in the C-suite or others in charge of the organization.

Spear-phishing attacks. Spear phishing refers to a specific type of targeted phishing attack. The attacker takes the time to research their intended targets and then write messages the target is likely to find personally relevant.

Ransomware. With Ransomware, the victim’s system is held hostage until they agree to pay a ransom to the attacker.

Password attacks. Passwords are the access verification tool of choice for most people, so figuring out a target’s password is an attractive proposition for a hacker.

SQL injection attacks. Structured Query Language (SQL) injection is a common method of taking advantage of websites that depend on databases to serve their users.

URL interpretation. With URL interpretation, attackers alter and fabricate certain URL addresses and use them to gain access to the target’s personal and professional data.

DNS spoofing. With Domain Name System (DNS) spoofing, a hacker alters DNS records to send traffic to a fake or “spoofed” website.

Session hijacking. Session hijacking is one of multiple types of MITM attacks. The attacker takes over a session between a client and the server.

Brute force attacks. A brute-force attack gets its name from the “brutish” or simple methodology employed by the attack. The attacker simply tries to guess the login credentials of someone with access to the target system.

Web attacks. Web attacks refer to threats that target vulnerabilities in web-based applications. Every time you enter information into a web application, you are initiating a command that generates a response.

Insider threats. Sometimes, the most dangerous actors come from within an organization. People within a company's own doors pose a special danger because they typically have access to a variety of systems.

Trojan horses. A Trojan horse attack uses a malicious program that is hidden inside a seemingly legitimate one. When the user executes the presumably innocent program, the malware inside the Trojan can be used to open a backdoor into the system.

Drive-by attacks. In a drive-by attack, a hacker embeds malicious code into an insecure website. When a user visits the site, the script is automatically executed on their computer, infecting it.

XSS attacks. With XSS, or cross-site scripting, the attacker transmits malicious scripts using clickable content that gets sent to the target's browser.

Eavesdropping attacks. Eavesdropping attacks involve the bad actor intercepting traffic as it is sent through the network.

Birthday attack. In a birthday attack, an attacker abuses a security feature: hash algorithms, which are used to verify the authenticity of messages.

Malware attack. Malware is a general term for malicious software, hence the "mal" at the start of the word. Malware infects a computer and changes how it functions, destroys data, or spies on the user or network traffic as it passes through.

Fighting Virus Sources in Distributed Networks Under SGT

Two simple examples are following.

(a) Nodes contain *records of being infected*, and also *from which neighbors*, as in Figure 8.

The following SGL scenario starting from any infected node (like "C") and tracing virus source via the infected predecessors may be as follows. This spatial cycle terminating in the virus source node having no registered infection predecessor, and its name is issued outside the network from the starting node.

```
hop_direct(C); STATUS == infected;
output(
  repeat(nonempty(Infected_from); hop(Infected_from));
  NAME)
```

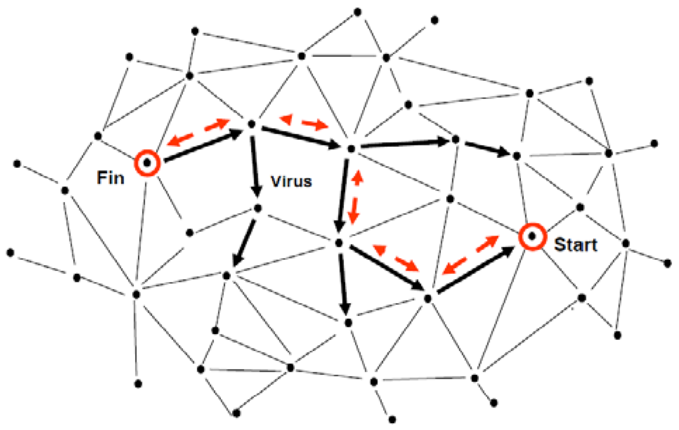

Figure 8. Finding virus source by moving to infected predecessor nodes.

(b) With the *infection time* registered in nodes, as in Figure 9.

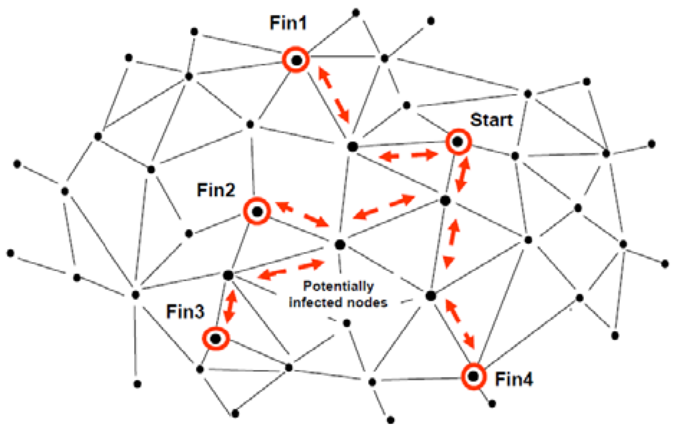

Figure 9. Finding most probable virus source via infection time in nodes.

Some explanation may be as follows. Starting from any nodes having infection time registered, the scenario first finds the neighboring nodes with earlier infection time and then moves to them. Repeating this networking search until such neighbors exist. This spatial process can potentially terminate in more than a single node, and the node with the earliest infection time is proclaimed either as virus source or be closest to it (as the latter may be self-protected from discovery).

```
hop_direct(any reachable nodes);
STATUS == infected; frontal(Time) = TIME;
output_min(
  repeat(
    hop(all_links; STATUS == infected; Infection_time < Time;
    Time = Infection_time);
  NAME && Time)
```

6. Comparison of Spatial Grasp Model With Traditional Algorithms

Algorithm Basics

Algorithm is a *finite sequence* of mathematically rigorous instructions typically used to solve a class of specific problems or to perform a computation (Wikipedia, 2025h). *A procedure for solving* a mathematical problem in a finite number of steps that frequently involves repetition of an operation (Merriam-Webster, Incorporated, 2025). *A procedure* used for solving a problem or performing a computation which acts *as an exact list of instructions* that conduct specified actions step by step in either hardware- or software-based routines (Gillis, 2024). A systematic procedure that produces in a *finite number of steps* the answer to a question or the solution of a problem (Britannica, 2025). A *set of mathematical instructions* or rules that, especially if given to a computer, helps to calculate an answer to a problem (Cambridge Dictionary, 2025). A *set of rules that must be followed* when solving a particular problem (Oxford Learner's Dictionaries, 2025). A *set of instructions* that is designed *to accomplish a task*, usually taking one or more inputs, running them systematically through a series of steps, and providing one or more outputs (National Library of Medicine, 2025). A *mathematical process* for solving a problem using a *finite number of steps*, being a key component of any computer program and the driving force behind various systems and applications (Nikolopoulou, 2023). A methodical, *step-by-step procedure* for solving problems or accomplishing tasks, acting as the backbone of software applications (HowStuffWorks, 2024).

Elementary algorithm structures are shown in Figures 10a and 10b, where Figure 10b reflects the flowchart with three decision-making nodes (1, 2, 3) for finding the greatest common divisor of two numbers (as from Wikipedia (2025h)).

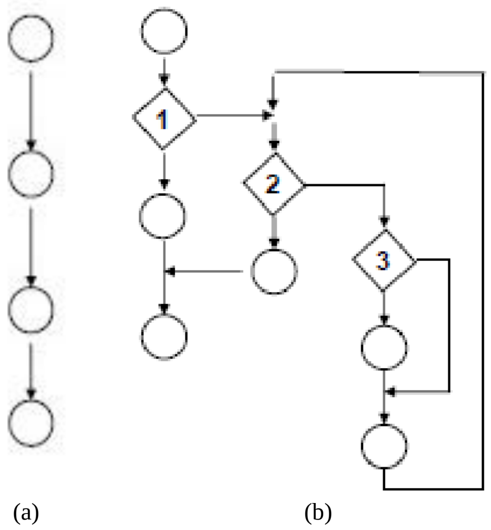

Figure 10. (a) elementary sequence of instructions, (b) with using branching and repetition.

Algorithm Extensions

Distributed algorithm is an algorithm designed to run on computer hardware constructed from *interconnected processors*; used in application areas of distributed computing (Wikipedia, 2025i). *Mobile agent* is a piece of software combined with data that is *able to migrate* from one computer to another autonomously and continue its execution on the destination (Wikipedia, 2025j). *Spatial analysis algorithms* are used for Geographic Information Systems (GIS) especially for *manipulation of map coordinates* (Lembo, n.d.).

In comparison with the described SG Model and Language, the traditional as well as extended algorithms may be considered *just as specific tools* for solving concrete problems. Whereas the Spatial Grasp may actually, not only symbolically, represent a universal *philosophy, paradigm and technology* oriented on explaining, covering, and ruling *the whole universe*.

7. Conclusions

The described Spatial Grasp paradigm covers any spatial systems philosophically, methodologically, and practically within the same spatial model and vision, whereas traditional algorithms being just specific tools for solving concrete problems.

This approach also fundamentally differs from any other models and languages for description, composition, control, and management of large distributed dynamic systems traditionally representing solutions by their distributed parts (or agents) communicating with each other and exchanging messages. Leaving this culture and types of operations only for the automatic implementation, the SGT scenarios describe all solutions on a much higher level, in the form of integral parallel spatial flooding, or even clever super-virus. The latter will always remain alive by being self-organized and self-confident during freely self-propagating, self-replicating, selfmatching, and self-recovering in distributed spaces, even after or during any possible disruptions and damages, in the most unpredictable and hostile environments. These high level spatial scenarios in SGL independently operating “over” rather than “in” distributed systems are much clearer and enormously shorter (up to hundred times) than in traditional C or Java languages.

Investigations and trial implementations of SGT and its predecessor WAVE were made in different countries and demonstrated via the internet (Sapaty, 1973; 1986; 1996; 1999; 2005; 2017; 2018; 2019; 2021; 2022; 2023; 2024a; 2024b; 2025; Bondarenko, Mikhalevich, Nikitin, & Sapaty, 1970). Any new technology versions can be carried out quickly by a small group of system programmers and on any existing platforms, also effectively integrated with advanced communication systems. Results of the ongoing investigation of SGT applicability in new areas of management, control, and cybersecurity (Cisco, 2025; Shutterstock, 2025; Fortinet, 2025) will also be revealed in the subsequent papers. The briefest technology reference: “Spatial Grasp” in google.com.

References

Bardt, C. (2024). *The feeling of space.* Cambridge: The MIT Press. Retrieved from https://mitpress.mit.edu/9780262049368/the-feeling-of-space/

Bondarenko, A. T., Mikhalevich, S. B., Nikitin, A. I., & Sapaty, P. S. (1970). Software of BESM-6 computer for communication with peripheral computers via telephone channels. In *Computer software* (Vol. 5). Kiev: Inst. of Cybernetics Press.

Britannica. (2025). Algorithm. Retrieved from https://www.britannica.com/science/algorithm

Cambridge Dictionary. (2025). Algorithm. Retrieved from https://dictionary.cambridge.org/dictionary/english/algorithm

Cisco. (2025). What is cybersecurity? Retrieved from https://www.cisco.com/site/us/en/learn/topics/security/what-is-cybersecurity.html

Corrigan, J. (2008). Spatiality and religion. In *The spatial turn.* London: Routledge. Retrieved from https://doi.org/10.4324/9780203891308

Dandoulaki, M., & Lazoglou, M. (2023). Disaster risk management and spatial planning: Evidence from the fire-stricken area of Mati, Greece. *Sustainability, 15*(12), 9776. Retrieved from https://doi.org/10.3390/su15129776

Fitzgibbons, L. (2019). Spatial intelligence. Retrieved from https://www.techtarget.com/whatis/definition/spatial-intelligence

Fortinet. (2025). Types of cyber attacks. Retrieved from https://www.fortinet.com/uk/resources/cyberglossary/types-of-cyber-attacks

Funk, G. A., Jansen, V. A. A., Bonhoeffer, S., & Killingback, T. (2005). Spatial models of virus-immune dynamics. *Journal of Theoretical Biology, 233*(2), 221-236. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0022519304004758

Galland, D., & Grønning, M. (2019). Spatial consciousness. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118568446.eurs0308

Geometriedidaktik. (2025). Spatial thinking. Retrieved from https://geometriedidaktik.at/en/spatial-ability/

Gillis, A. S. (2024). What is an algorithm? Retrieved from https://www.techtarget.com/whatis/definition/algorithm

Hipp, J. R. (2022). *The spatial scale of crime* (1st ed.). London: Routledge. Retrieved from https://www.amazon.com/SpatialScale-Crime-John-Hipp/dp/103220236X

Hopkins, P. (2022). *Afterword: Spatializing hate—Relational, intersectional and emotional approaches*. Bristol: Bristol University Press. doi:doi.org/10.56687/9781529215205-017

HowStuffWorks. (March 5, 2024). What is an algorithm? Retrieved from https://computer.howstuffworks.com/what-is-a-computer-algorithm.htm

IGI. (2025). What is spatial thinking? Retrieved from https://www.igi-global.com/dictionary/spatial-thinking/62031

Karmarkar, D. (2023). Spatial awareness and consciousness: Navigating life’s spaces. Retrieved from https://medium.com/@karmarkardipesh/spatial-awareness-and-consciousness-navigating-lifes-spaces54158e13a153

Kumberger, P., Frey, F., Schwarz, U. S., & Graw, F. (2016). Multiscale modeling of virus replication and spread. *Integrative Analysis of Pathogen Replication and Spread, 590*(13), 1972-1986. Retrieved from https://ouci.dntb.gov.ua/en/works/7BQBGPD9/

Laurini, R., & Thompson, D. (1992). Spatial knowledge: Intelligent spatial information systems. In *Fundamentals of spatial* *information systems* (pp. 620-670). New York: Academic Press. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/B9780080924205500220

Lembo, A. J. (n.d.). Spatial algorithms. Salisbury University. Retrieved from https://faculty.salisbury.edu/~ajlembo/419/lecture11.pdf

Merriam-Webster, Incorporated. (2025). Algorithm. Retrieved from https://www.merriam-webster.com/dictionary/algorithm

National Library of Medicine. (2025). Algorithm. Retrieved from https://www.nnlm.gov/guides/data-glossary/algorithm

Nejasmic, J., Bucher, L., & Knauff, M. (2015). Grounded spatial belief revision. *Acta Psychologica, 157*, 144-154. Retrieved from https://doi.org/10.1016/j.actpsy.2015.02.008

Nikolopoulou, K. (2023). What is an algorithm? Definition & examples. Retrieved from https://www.scribbr.com/aitools/what-is-an-algorithm/

Oliver, S. (2019). What is spatial temporal in philosophy? Retrieved from https://www.quora.com/What-is-spatial-temporal-inphilosophy#:~:text=This%20is%20an%20adjective%20describing,as%20your%20thoughts%20and%20emotions

Oxford Learner's Dictionaries. (2025). Algorithm. Retrieved from https://www.oxfordlearnersdictionaries.com/definition/english/algorithm

Roy, B. (2020). Space horizons: An era of hope in the geostationary orbit. *University of Oregon School of Law, 35*, 165-193. Retrieved from https://scholarsbank.uoregon.edu/items/dd40b459-2a12-4396-bdd8-e41348793bfe

Sapaty, P. S. (1973). A method of organization of an intercomputer dialogue in the radial computer systems. In *The design of software and hardware for automatic control systems*. Kiev: Inst. of Cybernetics Press.

Sapaty, P. S. (1986). A wave language for parallel processing of semantic networks. *Comput. Artif. Intell., 5*(4), 289-314.

Sapaty, P. S. (1993). A distributed processing system. European Patent N 0389655, Publ. 10.11.93, European Patent Office.

Sapaty, P. S. (1996). Distributed modeling of cooperative behavior by mobile agents. In *Proc. sixth conference on computer generated forces and behavioral representation, IST UCF* (pp. 599-613), Orlando, FL.

Sapaty, P. S. (1999). *Mobile processing in distributed and open environments.* New York: John Wiley & Sons.

Sapaty, P. S. (2005). *Ruling distributed dynamic worlds.* New York: John Wiley & Sons.

Sapaty, P. S. (2017). *Managing distributed dynamic systems with spatial grasp technology*. New York: Springer.

Sapaty, P. S. (2018). *Holistic analysis and management of distributed social systems*. New York: Springer.

Sapaty, P. S. (2019). *Complexity in international security: A holistic spatial approach*. London: Emerald Publishing,

Sapaty, P. S. (2021). *Symbiosis of real and simulated worlds under spatial grasp technology*. New York: Springer.

Sapaty, P. S. (2022). *Spatial grasp as a model for space-based control and management systems.* Boca Raton: CRC Press.

Sapaty, P. S. (2023). *The spatial grasp model: Applications and investigations of distributed dynamic worlds.* London: Emerald Publishing.

Sapaty, P. S. (2024a). *Providing integrity, awareness, and consciousness in distributed dynamic systems.* Boca Raton: CRC Press.

Sapaty, P. S. (2024b). *Spatial networking in the united physical, virtual, and mental world.* New York: Springer.

Sapaty, P. S. (2025). *Self-healing and self-recovering systems under the spatial grasp model.* London: Emerald Publishing Limited,

ScienceDirect. (1998). Spatial diversity. In *Wireless communications design handbook.* Retrieved from https://www.sciencedirect.com/topics/engineering/spatial-diversity

ScienceDirect. (2025). Spatial vision. Retrieved from https://www.sciencedirect.com/topics/neuroscience/spatial-vision

Seladi-Schulman, J. (2020). What's important about spatial awareness? Retrieved from https://www.healthline.com/health/spatial-awareness

Sellers, J. J., Astore, R. J., & Giffen, R. B. (2000). *Understanding space: An introduction to astronautics* (2nd ed.). New York: McGraw-Hill. Retrieved from https://www.amazon.com/Understanding-Space-Introduction-JerrySellers/dp/0072424680

Sellers, J. J., Astore, R. J., & Giffen, R. B. (2015). *Understanding space: An introduction to astronautics* (4th ed.). New York: McGraw-Hill Companies, Inc.

Shutterstock. (2025). Common internet cyber attacks. Retrieved from https://www.shutterstock.com/image-vector/vector-info-graphic-common-internetattacks-606244733

Twinkl. (2025). What is spatial awareness? Retrieved from https://www.twinkl.com/teaching-wiki/spatialawareness#:~:text=Spatial%20awareness%20is%20the%20ability,you%20can%20walk%20without%20bending

University of Passau. (2024). "Spatial beliefs" in the context of school. Multicollectivity, Education and Space. Retrieved from https://www.sobi.uni-passau.de/en/education-diversity-educational-spaces/research/education-and-space/translate-toenglisch-spatial-beliefs-im-kontext-schule

Wenger, M. R., & Lantz, B. (2022). Hate crime and place: The spatial and temporal concentration of bias-motivated crime in Washington, D.C. *Journal of Interpersonal Violence, 37*(13-14), 10683-10708. doi:10.1177/0886260520987817

Wikipedia. (2025a). Spacetime. Retrieved from https://en.wikipedia.org/wiki/Spacetime
Wikipedia. (2025b). Spatial analysis. Retrieved from https://en.wikipedia.org/wiki/Spatial_analysis
Wikipedia. (2025c). Spatial ability. Retrieved from https://en.wikipedia.org/wiki/Spatial_ability
Wikipedia. (2025d). Danger space. Retrieved from https://en.wikipedia.org/wiki/Danger_space
Wikipedia. (2025e). Space warfare. Retrieved from https://en.wikipedia.org/wiki/Space_warfare
Wikipedia. (2025f). Spatial planning. Retrieved from https://en.wikipedia.org/wiki/Spatial_planning
Wikipedia. (2025g). Religion in space. Retrieved from https://en.wikipedia.org/wiki/Religion_in_space
Wikipedia. (2025h). Algorithm. Retrieved from https://en.wikipedia.org/wiki/ Algorithm
Wikipedia. (2025i). Distributed algorithm. Retrieved from https://en.wikipedia.org/wiki/Distributed_algorithm
Wikipedia. (2025j). Mobile agent. Retrieved from https://en.wikipedia.org/wiki/Mobile_agent

Appendix: Spatial Grasp Language Summary

In the language summary below, syntactic categories are shown in italics, vertical bar separates alternatives, parts in braces indicate zero or more repetitions with a delimiter at the right, and constructs in brackets are optional. The remaining characters and words are the language symbols (including boldfaced braces).

grasp	→	*constant* | *variable* | [*rule*] [(**{** *grasp*,**}**)]
constant	→	*information* | *matter* | *special*
information	→	*string* | *scenario* | *number*
string	→	‘{*character*}’
scenario	→	**{**{*character*}**}**
number	→	[*sign*]{*digit*}[.{*digit*}[e[*sign*]{*digit*}]]
matter	→	“{*character*}”
special	→	thru | done | fail | fatal | infinite | nil | any | all | other | allother | current | passed | existing | neighbors | direct | forward | backward | synchronous | asynchronous | virtual | physical | executive | engaged | vacant | firstcome | unique
variable	→	*global* | *heritable* | *frontal* | *nodal* | *environmental*
global	→	G{*alphameric*}
heritable	→	H{*alphameric*}
frontal	→	F{*alphameric*}
nodal	→	N{*alphameric*}
environmental	→	TYPE | NAME | CONTENT | ADDRESS | QUALITIES | WHERE | BACK | PREVIOUS | PREDECESSOR | DOER | RESOURCES | LINK | DIRECTION | WHEN | TIME | STATE | VALUE | IDENTITY | IN | OUT | STATUS | SIZE | WEIGHT | LENGTH
rule	→	*movement* | *creation* | *advancement* | *branching* | *cycling* | *echoing* | *verification* | *assignment* | *transference* | *exchange* | *timing* | *qualifying* | *type* | *usage*
movement	→	hop | hopfirst | hopforth | move | shift | follow
creation	→	create | linkup | delete | unlink
advancement	→	advance | slide | align | fringe
branching	→	branch | sequence | parallel | if | or | and | orsequence | orparallel | andsequence | andparallel | choose | quickest | | split | replicate
cycling	→	repeat | cycle | loop | sling | whirl
echoing	→	state | rake | order | unit | unique | sum | count | first | last | min | max | random | average | sortup | sortdown | reverse | element | position | fromto | add | subtract | multiply | divide | degree | separate | unite | attach | append | common | withdraw | increment | decrement | access | invert | apply | location | distance
verification	→	equal | nonequal | less | lessorequal | more | moreorequal | bigger | smaller | heavier | lighter | longer | shorter | empty | nonempty | belong | notbelong | intersect | notintersect | yes | no
assignment	→	assign | assignpeers
transference	→	run | call
exchange	→	input | output | send | receive | emit | get
timing	→	sleep | allowed
qualifying	→	contain | release | trackless | free | blind | quit | abort | stay | lift | seize | exit
type	→	global | heritable | frontal | nodal | environmental | matter | number | string | scenario | constant
usage	→	address | coordinate | content | index | time | speed | name | place | center | range | doer | node | link | unit