WAVE-WP (WORLD PROCESSING) TECHNOLOGY

Peter Sapaty [1] Masanori Sugisaka [2]

[1] *Institute of Mathematical Machines and Systems*
National Academy of Sciences,
Glushkova Ave 42, 03187 Kiev, Ukraine

[2] *Department of Electrical and Electronic Engineering*
Oita University, 700 Oaza Dannoharu, 870-1192 Japan

Keywords: parallel and distributed processing, distributed control, space navigation, spatial pattern-matching, WAVE-WP language, system integrity, system management, distributed simulation, cooperative robotics, open systems.

Abstract. A new computational and control model of parallel and distributed nature has been developed. It comprises self-evolving, space-conquering automaton, high-level system navigation and coordination language, describing system problems in a spatial pattern-matching mode, and related distributed control mechanisms for management of physical, virtual, and combined worlds. The model allows us to obtain complex spatial solutions in a compact, integral, and seamless way. It can be effectively used for the creation, integration, simulation, processing, management and control of a variety of dynamic and open systems – from physical to biological, and from artificial to natural.

1 INTRODUCTION

The use of computers is steadily shifting from computations to coordination and management of complex distributed and dynamic systems, in both civil and military areas. To speed up the progress in this direction, we need qualitatively different knowledge processing and control models as traditional ones, like Turing machine or cellular automata, underlying conventional computers and programming in them, were originally oriented on computations. We also need models that can describe much broader activities than traditional information processing, to operate in real worlds, with physical movement and physical matter and objects relocation and manipulation.

The execution of such models may need to involve any existing manned or unmanned hardware and software systems in the world, human beings including. A possible general-purpose model of this type, called WAVE-WP (or World Processing), as a further development of the distributed network processing WAVE paradigm (Sapaty, 1999), will be summarized in this paper.

2 PROBLEMS OF DISTRIBUTED PROCESSING AND CONTROL TECHNOLOGIES

Single machine solutions are often showing highest possible integrity as a system – with all resources directly available, and control being global, direct, and absolute. On the contrary, the existing distributed computing and control models and technologies follow the analytical approach, representing and studying systems as consisting from pieces, or *agents*, exchanging messages or sharing common (distributed) objects. This results in known difficulties of obtaining the needed global behavior from local activities, especially if there many of them and their number and interactions change over time.

Sapaty P. and Sugisaka M. (2004).
WAVE-WP (WORLD PROCESSING) TECHNOLOGY.
In *Proceedings of the First International Conference on Informatics in Control, Automation and Robotics*, pages 92-102
DOI: 10.5220/0001152400920102

Overhead in making large distributed systems operate as a single controllable entity and follow global goals, by using such approaches, may be enormous, as symbolically shown in Fig. 1, see also (Sapaty, 2002), and the necessity of runtime recovery after failures may aggravate the situation further.

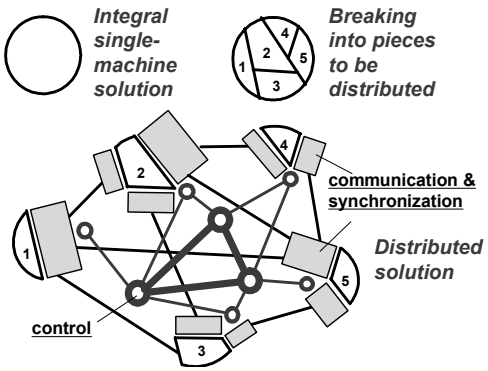


Figure 1: Overhead of distributed system solutions.

For solving complex problems in real, especially unpredictable and hostile environments, we may need distributed systems of much higher integrity, being based on quite different (than traditional pieces-to-whole) ideologies and methodologies.

The WAVE-WP model tries to attack the problem by starting from the opposite side – from the whole, allowing us to express it directly, while abstracting from possible system's parts and their interactions, delegating the latter to an efficient automatic implementation.

3 THE WAVE-WP CONTROL AUTOMATON

The automaton effectively inherits the integrity of traditional sequential programming over localized memory, but for working now with the real distributed world, while allowing its parallel navigation in an active pattern flow and matching mode – as a single spatial process.

The automaton may start from any point of the distributed world or system to be controlled, dynamically covering its parts or the whole, and mounting of a variety of parallel and distributed knowledge and control infrastructures. Implanting distributed "soul" into the system organization, the automaton increases the system's integrity, capability of pursuing local and global goals, assessing distributed situations, making autonomous decisions, and recovering from indiscriminate damages.

In quite other applications, it may need to destroy the very system it propagates through and/or operates in (or certain, for example, malicious incursions or infrastructures in it). Many spatially cooperating or competing parallel WAVE-WP automata may evolve on the same system's body serving, say, as deliberative, reactive, and/or reflective spatial processes.

World representation. One of the main features of WAVE-WP is the representation of distributed worlds it operates in, as described in (Sapaty, 2000).

Physical world (or PW) is continuous and infinite in WAVE-WP. Existing at any its point, and possibly performing a job, is considered as residing in a *node* (with physical coordinates). Such a node, reflecting only occupancy at the point, has no personal identity or content. It vanishes with the termination of all occupancies in it.

Virtual world (or VW) is discrete and interlinked in WAVE-WP, and is represented, similar to WAVE (Sapaty, 1999), by a distributed Knowledge Network (KN). Its persistent nodes may contain established concepts or facts, and (also persistent) links (oriented and non-oriented, connecting the nodes) may reflect different relations between the nodes.

The same model can also operate with the *united (or PVW) world*, in which any element may have features of the both worlds. A simplified example of such a world is shown in Fig. 2.

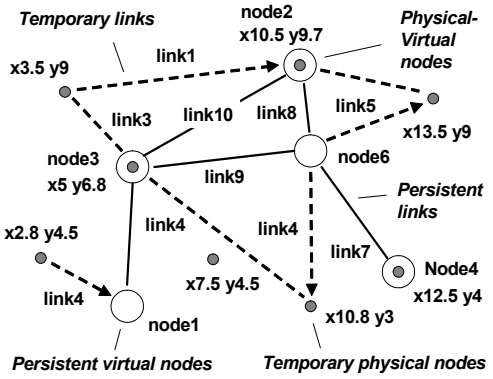


Figure 2: United distributed physical-virtual world.

A variety of effective access mechanisms to nodes, links and their groups, say, by physical coordinates, electronic addresses, by names, via traversing links, etc. (classified as *tunnel* and *surface* navigation) abound in the model, using both selective and broadcasting access modes.

Waves. Solutions of any problems in this formalized world in WAVE-WP are represented as its coordinated parallel *navigation* (or *exploration*, *invasion*, *grasping*, *coverage*, *flooding*, *conquest*, etc.) by some higher-level forces, or *waves* (Sapaty, 1999, 2000, 2002). These bring local operations, control and transitional data directly into the needed points of the world, to perform jobs there. The obtained results, together with the same or other operations may, in their turn, invade the other world parts, and so on. In a most abstract form this "agent-less" spatial process may look like shown in Fig. 3, with parallel asynchronous navigation, spatial intersection, branching and looping.

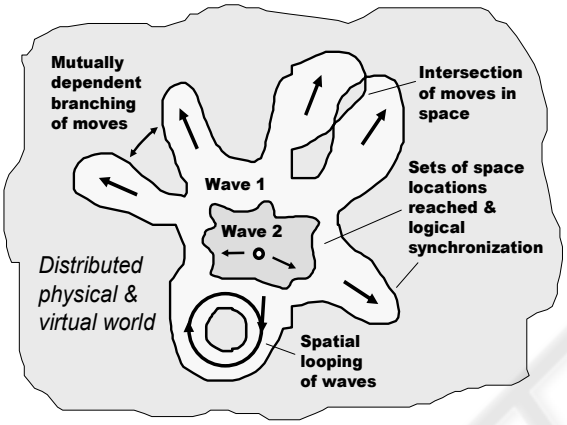


Figure 3: Parallel conquest of distributed worlds by waves.

During the world navigation, which may be loose and free or strictly (both hierarchically and horizontally) controlled, waves can modify the very world they evolve in and move through, as well as create it from scratch (including any distributed structures and topologies). Waves may also settle persistent cooperative processes in its different points, subsequently influencing, *governing* its further development and evolution in the way required.

4 HIGHER-LEVEL WAVE-WP LANGUAGE

The system language expressing full details of this new control automaton has been developed. Having recursive space-navigating and space-penetrating nature, it can operate with both information and physical matter. The language can also be used as a traditional one, so no integration with (and/or interfaces to) other programming models and systems may be needed for solving complex distributed knowledge processing and control problems.

Very compact syntax of the language, as shown in Fig.4 (see also Sapaty, Sugisaka, 2002), makes it particularly suitable for direct interpretation in distributed environments, being supported by effective program code mobility in computer networks. In this description, braces set up zero or more repetitions of a construct with a delimiter at its right; square brackets identify an optional construct; semicolon allows for sequential, while comma for parallel invocation of program parts; and parentheses are used for structuring of WAVE-WP programs (or *waves*).

Successive program parts, or *advances*, develop from all nodes of the set of nodes reached (SNR) by the previous advance, whereas parallel or independent parts, *moves*, constituting the advances, develop from the same nodes, while splitting processes and adding their own SNRs to the resultant SNR of the advance.

wave	→	**{ *advance* ; }**
advance	→	**{ *move* , }**
move	→	***constant*** **|** ***variable*** **| { *move act* }	[*rule*] (*wave*)**
constant	→	***information*** **|** ***physical-matter***
variable	→	***nodal*** **|** ***frontal*** **|** ***environmental***
act	→	***flow-act*** **|** ***fusion-act***
rule	→	***forward-rule*** **|** ***echo-rule***

Figure 4: Syntax of WAVE-WP language.

Elementary *acts* represent data processing, hops in both physical and virtual spaces, and local control. *Rules* establish non-local constraints and contexts over space-evolving waves like, for example, the ability to create networks, also allowing WAVE-WP to be used as a conventional language. *Variables*, called *spatial* (as being scattered in space), can be of the three types: *nodal,* associated with virtual or physical nodes and shared by different waves; *frontal,* propagating with waves as their sole property; and *environmental,* accessing elements of internal and external environments navigated by waves.

This recursive navigational structure of the language allows us to express highly parallel and fully decentralized, albeit strongly controlled and coordinated, operations in distributed worlds in a most compact way – in the form of integral space processing and transformation formulae. These formulae resemble data processing expressions of traditional programming languages, but can now *operate in and process the whole distributed world*.

5 IMPLEMENTATION BASICS

On the implementation layer, the automaton widely uses high-level mobile cooperative program code self-spreading and replicating in networks, and can be easily implemented on any existing software or hardware platform. As the automaton can describe direct movement and processing in physical world, its implementation may need to involve a multiple mobile hardware – with or without human participation. A network of (hardware or software) communicating WAVE-WP language interpreters (or WI), which can be mobile if installed in manned or unmanned vehicles, should be embedded into the distributed world to be controlled (Sapaty, 1999, 2002), placing WIs in world's most sensitive points, as shown in Fig. 5.

During the spatial execution of system scenarios in WAVE-WP, individual interpreters can make local information and physical matter processing as well as physical movement in space. They can also partition, modify and replicate program code, sending it to other interpreters (along with local transitional data), dynamically forming *track-based distributed interpretation infrastructures*.

The automaton can also exploit other systems as computational and control resources, with or without preliminary consent, i.e. in a (remotely controlled) virus-like mode. For example, existing network attacks (especially DDoS) may be considered as a possible malicious (simplified, degenerated, and distorted) implementation of the automaton.

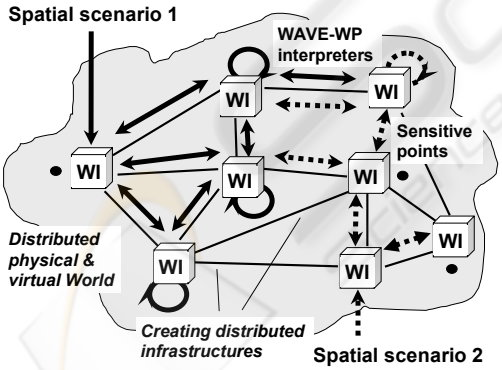


Figure 5: A network of interpreters with self-spreading spatial scenarios in WAVE-WP.

6 THE DISTRIBUTED WAVE-WP INTERPRETER

The WAVE-WP interpreter consists of a number of specialized processors working asynchronously and in parallel, handling and sharing specific data structures like waves queue, incoming and outgoing queues, local part of the distributed knowledge network, track forest, etc., and being responsible for different interpretation operations (parser, data processor, control processor, communication processor, sensors and motion, etc.), see Fig. 6 (also Borst, 2002; Sapaty, 1993, 1999).

The interpreter can be easily implemented on any existing platform, in software or directly in silicon. The existing public domain WAVE system (in C under Unix/Solaris/linux) operating via the Internet had been used in different countries, especially for distributed network management (Gonzalez-Valenzuela, Vuong, 2002) and simulation of battlefields (Sapaty, 1999, 2002).

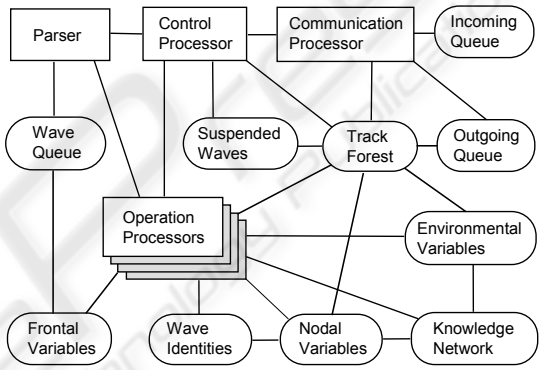


Figure 6: General organization of WAVE-WP interpreter

The interpreter may also have physical body, say, as a mobile or humanoid robot, if engaged in operations in physical world, or can be body-mounted on humans. The whole network of such "doers" can be mobile, changing structure, as robots or humans can be moving at runtime (Sapaty, Sugisaka, 2002a). Elementary expressions and operations of WAVE-WP language may trigger a combination of physical movements of doers in space, transference of information and physical matter between the doers both electronically, at a distance, and in a direct, tactile contact, as well as cooperation and group behavior of doers.

The following program first hops into some node `a` of the distributed KN (located, say, in `Doer.1`), picks up a certain matter from the environment there into a frontal variable `F`, also information (integer `5`) into `F1`, then propagates via virtual link `p` to another node `b` (which may happen to reside in `Doer.2`), bringing the matter and information into the latter doer. The rest of program (or `wave`) will be performed from the destination doer.

```
direct#a; F="matter"; F1=5; p#b; wave
```

Coming to node `b` with `F1` and `wave` could be done purely wirelessly, but the transference of `F` needs direct physical contact, which can be performed by either `Doer.1` or `Doer.2` (or both) moving for a meeting, or by engagement of another doer, say `Doer.3`, for the matter's transference, as shown in Fig. 7.

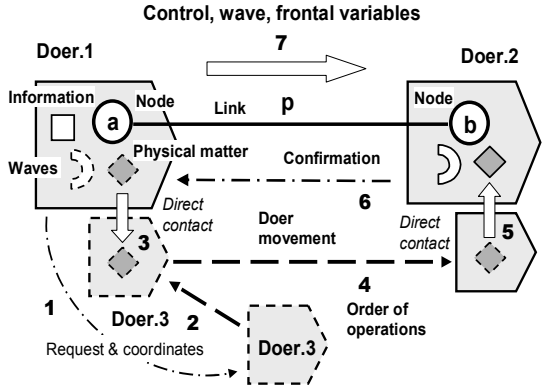


Figure 7: Transference of physical matter between doers

7 DIRECT ACCESS TO PHYSICAL WORLD

The model can describe complex operations in PW on a semantic level, abstracting from implementation, with parallel control directly evolving in physical space, covering the latter. Multiple operations can be performed cooperatively by groups of robots or humans, with transfers of information and physical matter between the interpreters embedded into physical bodies, communicating both electronically and mechanically. A coordinated delivery of physical matter to remote points and its synchronized processing by a group of robots had been demonstrated (Sapaty, Sugisaka, 2002a). A description of coordinated robotic column movement and its synchronized operation had been shown (Sapaty, Sugisaka, 2001).

As an elementary example of direct operations in PW, the following program, applied in a certain workplace, brings from remote locations sand, cement and water, mixing them and making concrete, and then delivers the obtained result into another remote location (assigning it to nodal variable `Nconcrete` there), using spatial programs `w1-w4` for accessing the mentioned locations physically (see Fig. 8):

```
(W4; Nconcrete) =
(w1; "3 tons of sand") +
(w2; "2 tons of cement") +
(w3; "4 tons of water")
```

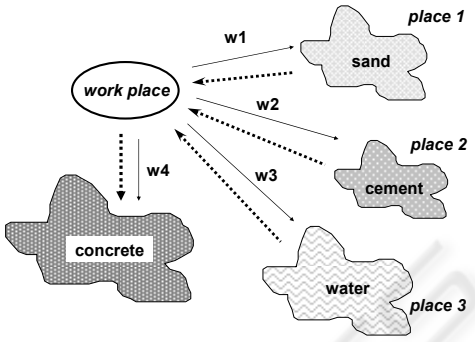


Figure 8: Direct operations on physical matter.

8 DISTRIBUTED KNOWLEDGE REPRESENTATION AND PROCESSING

Dynamically creating arbitrary knowledge networks in distributed spaces, which can be modified at runtime, WAVE-WP can implement any knowledge processing and control systems in parallel and fully distributed way, similar to WAVE (Sapaty, 1999). A program package in WAVE was demonstrated for basic problems of the graph and network theory, where each graph node can reside on a separate computer. The package included finding spanning trees, shortest paths, articulation points, maximum cliques, diameter and radius, etc., also self-recovering network topologies after indiscriminate damages of their nodes and links. Parallel simulation of other control models in WAVE-WP, like Petri nets, was demonstrated too.

As an example, let us consider a KN of Fig. 9, which may be arbitrarily distributed between computers of Internet or robots, say, `Robot.1` and `Robot.2`.

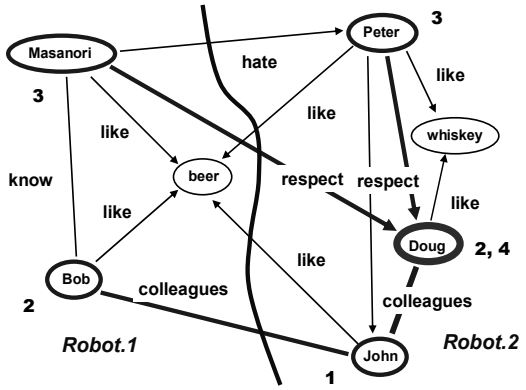


Figure 9: Distributed knowledge representation.

In WAVE-WP, complex knowledge processing tasks can be formulated and solved in parallel, regardless of the knowledge distribution. Say, to answer the question: "Who of John's colleagues is respected by both Masanori and Peter", the following simple program will be sufficient:

```
direct#John; colleagues#any;
and(
  andparallel(-respect#(Masanori, Peter)),
  USER = CONTENT)
```

It provides the final answer `Doug`, with the decision-finding process shown in bold and stages numbered in Fig. 9 (parallel operations emerging on stage 3).

9 OPERATING IN PHYSICAL WORLD UNDER THE GUIDANCE OF VIRTUAL WORLD

Operating in the unity of physical and virtual worlds, the WAVE-WP model can effectively investigate physical worlds and create their reflection in the form of distributed virtual worlds. The latter can guide further movement and search operations in the distributed PW, modifying and updating itself, and so on. Such physical-virtual world unity in one model, where both worlds can be open and dynamic, allows us to create integral intelligent manned or unmanned systems operating effectively in unpredictable environments.

This has been demonstrated on an example of optimized parallel territory search by a group of robots sharing distributed knowledge representation of the area of interest, which can be modified and updated (say, reduced or extended) at runtime (Sapaty, Sugisaka, 2003). Parallel operations in PW can be optimized in advance by solving the problems in VW first, in a parallel simulation mode, mapping subsequently or simultaneously the obtained solutions into the PW.

For example, the space to be searched can be described by a set of connected polygons and represented in WAVE-WP by a KN reflecting the polygons connectivity graph and individual polygon data, as shown in Fig. 10. This spatially shared KN can be arbitrarily distributed between robots and constantly updated by them in different points.

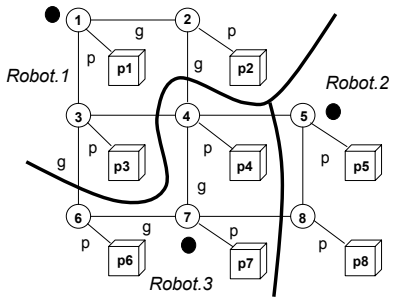


Figure 10: Space model distribution between robots

For example, by visiting polygons only once and moving to free neighboring polygons, while checking the states of surrounding neighbors sequentially, the program, consisting of three competing branches starting in polygons `1`, `5`, and `7` by different robots, will be as follows (using specific external procedure `move_clean`):

```
direct#(1,5,7); M=1; (p#)?move_clean;
repeat(or(g#; grasp(M==nil; M=1));
       (p#)?move_clean)
```

10 INTELLIGENT NETWORK MANAGEMENT

Integrating traditional network management tools and systems, and dynamically extracting higher-level knowledge from raw data via them, WAVE-WP, same as WAVE, establishes a higher, intelligent layer, allowing us to analyze varying network topologies, regulate network load, and redirect traffic in case of line failures or congestions (Sapaty, Zorn, 1991). Representing a universal spatial control model based on coordinated code mobility and dynamic tracking, WAVE-WP can also be used for creating essentially new, universal and intelligent, network protocols. These, along with traditional data delivery, will be able to make local and global automatic decisions on the network management, self-recovery after failures, and runtime topology restructuring and optimization.

As the simplest example for distributed network management in WAVE-WP, the following maximum parallel program creates shortest path tree (SPT) from some node `a`, covering the whole network in a spatial navigation mode, by self-replicating mobile code (see also Sapaty, 1999):

```
direct#a;
repeat(#;Fd+=L;Nd==,Nd>Fd;Nd=Fd;Np=B)
```

Another program, starting from node `e`, collects the shortest path from `a` to `e` in a reverse order via the SPT found and recorded in a computer network by the previous program:

```
direct#e;repeat(Fp&=C;#Np);U=Fp
```

The work of these two programs is shown in Fig. 11.

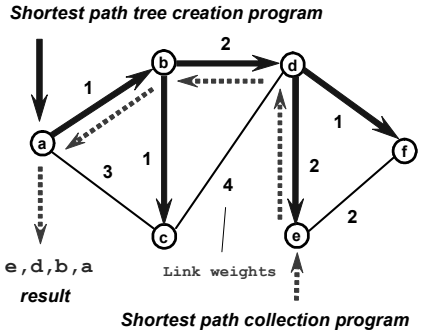


Figure 11: Distributed dynamic shortest path solution.

11 ADVANCED CRISIS REACTION FORCES

Smaller, dynamic armies, with dramatically increased mobility and lethality, represent nowadays the main direction, also challenge, in the development of advanced military forces oriented on crisis situations. These forces should be capable of conducting non-traditional combat, peacekeeping, and recovery operations, withstanding asymmetric threats, and operating in unpredictable environments. They may also effectively use multiple unmanned units, such as mobile robots.

The WAVE-WP technology can quickly assemble a highly operational battle force from dissimilar (and possibly casual) units, setting intelligent command and control (CC) infrastructures over them (Sapaty, 2000a). These infrastructures can follow global mission scenarios and make autonomous decisions in manned, automated, or fully unmanned mode. In case of damages, the technology can either restore the previous CC infrastructure (with, possibly, reduced set of units), or make complete runtime reassembling of the entire force, with a new CC infrastructure, taking into account remaining operational units and their physical locations. Both local and global restructuring and reassembling can be carried out at runtime, without the loss of overall operational capability.

As an example, the following parallel program creates and regularly updates an optimal hierarchical CC infrastructure based on a physical neighborhood, with top of the hierarchy always associated with the most central unit of a distributed force. Individual units (manned or unmanned vehicles, computerized humans) can be on a constant move, but the optimal spatial CC hierarchy is maintained under any circumstances (including indiscriminate destruction of units).

```
repeat(
  Faver=average(direct#all; WHERE);
  Nstart=min(direct#all;
    (Faver, WHERE)?distance_ADDRESS):2;
  direct#Nstart;
  quit(
    Frange=r40; N=1;
    repeat(
      direct#Frange; grasp(N==nil; N=1);
      [any#any; LINK=nil];
      [create(-infra#BACK)]));
  USER=ADDRESS; TIME+=360)
```

A possible hierarchical infrastructure, created and maintained by this program, is shown in Fig. 12.

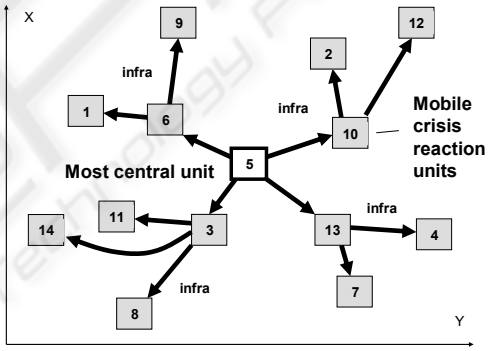


Figure 12: Creating of a neighborhood hierarchy

12 MASSIVE COOPERATIVE ROBOTICS

Autonomous robotic armies for civil and military applications, comprising thousands of cooperating mobile units, may well become today's reality in WAVE-WP (Sapaty, Sugisaka, 2002a). High-level mission scenarios can be injected into such organizations from any robotic unit, dynamically covering and grasping the whole distributed system.

Massive coordinated operations in physical spaces can automatically emerge during parallel interpretation of the WAVE-WP scenarios, while preserving the overall system integrity and ability of pursuing global goals. The scenarios can be represented in such a way that they should (and will be able to) survive by any means, while individual

robots may fail, with system intelligence being the feature of the whole campaign, rather than of individual robots.

The following program sets up a number of cyclic routes (with waypoints given) to be patrolled, and then makes parallel branches compete and seize available robotic bodies themselves, to follow these routes. If a robot finds some irregularities on its own route (using infrared sensors), it reports this to the user of the whole system, and also reprograms other robots in order to follow its own route for some period of time. After the expiration of this time, these other robots return to patrol their own routes. The work of this program (for only two robots supposedly available) is depicted in Fig. 13.

```
Fpoints=(x4y3,x7y3,x9y7,x5y10,x2y6),
Fpoints=(x11y4,x11y8,x13y10,x15y8,x14y3),
.............................;
or(direct#all; grasp(Nmark==; Nmark=1));
Np=Fpoints;
repeat(
 WHERE=Np:1; Np&=Np:1; Np:1=nil; Fp=Np;
 free(?infraredCheck; USER=alarm_WHERE;
      direct#any; Np=Fp; TIME+=1800;
      Np=Fpoints))
```

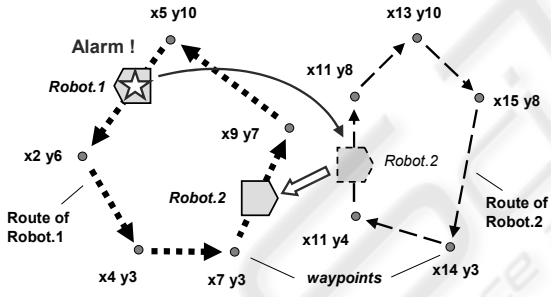


Figure 13: Cooperative region patrol by two mobile robots

13 DISTRIBUTED ROAD AND AIR TRAFFIC MANAGEMENT

Distributed computer networks working in WAVE-WP and covering the space to be controlled can be efficiently used for both road and air traffic management. The WAVE model provides simultaneous tracking of multiple objects in PW by mobile intelligence spreading in the VW, via computer networks (Sapaty, Corbin, Seidensticker, 1996; Sapaty, 1999; Sapaty, Klimenko, Sugisaka, 2004).

Assigning personal active mobile code to each object under control brings high flexibility to the distributed control system, with parallel and cooperative tracking of multiple objects and making non-local decisions, along with runtime optimization and routing. This may especially be important in crisis situations, where a priori flight schedules become useless, and the distributed management infrastructure is damaged. The surface roads can be destroyed too, and traffic routing may need to be fully dynamic, in order to reach destinations by individual vehicles in suitable time. Related distributed management scenarios with mobile intelligence had been demonstrated live in WAVE via the Internet in different projects.

As an elementary example, the following simple mobile program (using an external procedure `Fobject?seen`) seizes control of a certain moving object, and then follows its PW movement, propagating itself via the computer network. It launches subordinate search agents in neighboring computers (associated, say with neighboring radar stations) unless the disappeared object is found elsewhere and followed again. The work of this program is shown in Fig. 14.

```
Fobject=TRW562;
repeat(
  repeat(Fobject?seen; TIME+=10);
   any#any; Fobject?seen)
```

Many such objects can be simultaneously seized and controlled by mobile intelligences in WAVE-WP, and any payload can be added to the program above, including detailed studying of the behavior of controlled objects, with their possible subsequent rerouting or destruction.

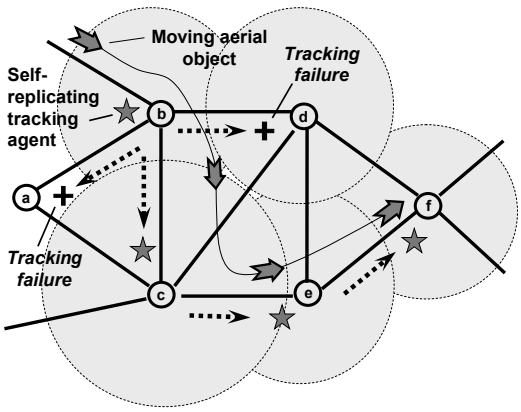


Figure 14: Distributed tracking of moving objects.

14 AUTONOMOUS DISTRIBUTED COGNITIVE SYSTEMS

Cognitive systems belong to the most advanced class of intelligent systems -- the ones aware of what they are doing. While cognitive systems include reactive and deliberative processes, they also incorporate mechanisms for self-reflection and adaptive self-modification.

The WAVE-WP paradigm allows for the description of interacting deliberative, reactive, and reflective processes on a semantic level, representing the whole mission rather than individual robots (Sapaty, Kawamura, Sugisaka, Finkelstein, 2004). This provides new degrees of freedom for autonomous robotic teams, where collective behavior of robots emerges as a derivative of parallel and distributed interpretation of WAVE-WP language, in the united physical and virtual world.

The mission-to-hardware mapping process may be fully distributed, not requiring central resources, and each robot may happen to be involved at any level of the distributed command and control process in any moment of time, as shown in Fig. 15. Failed robots can be automatically substituted at runtime without loss of the overall mission integrity.

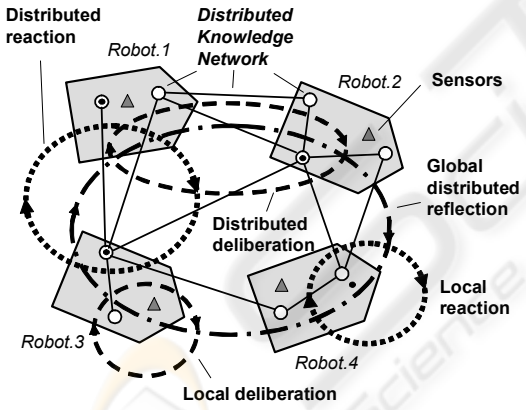


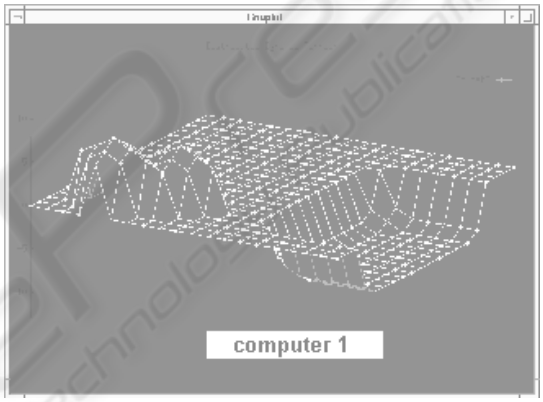
Figure 15: Fully distributed cognitive system.

15 DISTRIBUTED INTERACTIVE SIMULATION

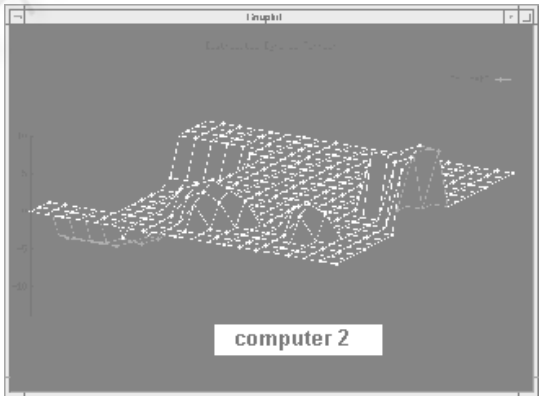
Having full control over distributed worlds, including their runtime creation and modification, WAVE-WP allows for highly efficient, scalable, distributed simulation of complex dynamic systems, like battlefields, in open computer networks, using potentially unlimited number of computers working together (Sapaty, Corbin, Borst, 1995).

Due to full distribution of the simulated space and entities operating in it, there is no need to broadcast changes in terrain or positions of moving entities to other computers, as usual. Each simulated entity operates in its own (current) part of the simulated world, within the range of its sensors, communicating locally with other such entities, exactly as in the real world. The entities can move freely through the simulated space (and between computers, if needed).

Using volatile virus-like spatial algorithms in WAVE-WP, a fully dynamic terrain can be effectively modeled in a distributed space. Its parts like clouds, floods, smog, mountains, landslides, and craters can grow, move and spread seamlessly between computers (Darling, Sapaty, Underhill, 1996; Sapaty, 1999), as shown in Fig. 16a,b.



a)



b)

Figure 16: Seamless distributed dynamic terrain modeling.

16 INTELLIGENT GLOBAL DEFENSE AND SECURITY INFRASTRUCTURES

WAVE-WP can also be effectively used in a much broader scale, especially for the creation of intelligent national and international infrastructures of different natures, widely using automated and fully automatic control and advanced robotics. Such global systems may effectively solve the problems of distributed air defense, where multiple hostile objects penetrating country's air space can be simultaneously discovered, chased, analyzed, and destroyed using a computerized networked radar system as a collective artificial brain operating in WAVE-WP, as shown in Fig. 17 (see also Sapaty, Klimenko, Sugisaka, 2004).

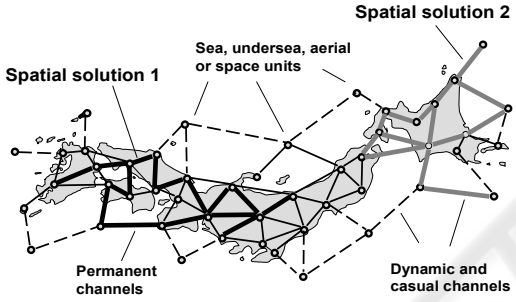


Figure 17: Distributed infrastructures and spatial solutions

Within the unified command and control infrastructures, provided by the technology, different types of unmanned air vehicles may be used, for example, as possible mobile sensor, relay, or even air traffic management stations, supplementary to the ground ones, especially when the latter get damaged or operate partially.

In other non-local applications, WAVE-WP, using worldwide computer networks, may effectively discover and trace criminals and their distributed organizations, penetrate into malicious infrastructures, studying and eliminating them, with possible additional involvement of special hardware and troops (Sapaty, 2002).

17 CONCLUSIONS

WAVE-WP allows for a more rational and universal integration, management, simulation, and recovery of large complex systems than many other approaches – by establishing a higher level of their vision and coordination, symbolically called "over-operability" (Sapaty, 2002) versus (and in supplement to) the traditional "interoperability".

Distributed system management and coordination scenarios in WAVE-WP are often orders of magnitude simpler and more compact than usual, due to high level and spatial nature of the model and language. They often help us to see the systems and solutions in them as a whole, avoiding tedious partitioning into parts (agents) and setting their communication and synchronization.

These and other routines are effectively shifted to the efficient automatic implementation by dynamic networks of WAVE-WP interpreters. Traditional software and hardware agents are being requested, created, and have sense only when required in certain moments of time, during the spatial development of self-evolving (conceptually agent-less) parallel mission scenarios.

A detailed description of the WAVE-WP model and its extended applications can soon be available (Sapaty, 2004).

REFERENCES

Borst, P.M., 2002. *An architecture for distributed interpretation of mobile programs*. PhD Thesis, also published as a book, ISBN 3-8316-0103-8.

Darling, J.C.C., Sapaty, P.S., Underhill, M.J., 1996. Distributed virtual reality: a fully dynamic approach. In *Proc. 15th Workshop on Standards for the Interoperability of Distributed Simulations*. IST UCF, Orlando, FL.

Gonzalez-Valenzuela, S., Vuong, S.T., 2002. Evaluation of migration strategies for mobile agents in network routing. In *Proc. The 4th International Workshop MATA'02*, Barcelona, Spain.

Sapaty, P.S., Zorn W., 1991. The WAVE model for parallel processing and its application to computer network management. In *International Networking Conference INET'91*, Copenhagen.

Sapaty, P.S., 1993. *A distributed processing system*. European Patent No. 0389655, European Patent Office.

Sapaty, P.S., Corbin, M.J., Borst, P.M., 1995. Towards the development of large-scale distributed simulations. In *Proc. 12th Workshop on Standards for the Interoperability of Distributed Simulations*. IST UCF, Orlando, FL.

Sapaty, P.S., Corbin, M.J., Seidensticker, S., 1996. Mobile intelligence in distributed simulations. In *Proc. 14th Workshop on Standards for the Interoperability of Distributed Simulations*. IST UCF, Orlando, FL.

Sapaty, P.S., 1999. *Mobile processing in distributed and open environments*, John Wiley & Sons. ISBN: 0471195723, New York.

Sapaty, P.S., 2000. High-level spatial scenarios in WAVE. In *Proc. The Fifth International Symposium on Artificial Life and robotics, (AROB 5th'00),* Beppu, Japan.

Sapaty, P.S., 2000a. Basic distributed control model and technology for mobile crisis reaction forces and their united air defense. In *Proc. NATO Symosium on System Concepts for Integrated Air Defense of Multinational Mobile Crisis Reaction Forces*. Valencia, Spain.

Sapaty, P., Sugisaka, M., 2001. Towards the distributed brain for collectively behaving robots. In *Proc. International Conference on Control, Automation and Systems, ICCAS*. Cheju National University, Jeju Island, Korea.

Sapaty, P.S., 2002. Over-operability in distributed simulation and control. *The MSIAC's M&S Journal* Online. Winter Issue, Volume 4, No. 2, Alexandria, VA, http://www.msiac.dmso.mil/journal/WI03/sap42_1.html.

Sapaty, P.S., Sugisaka, M., 2002. A Language for programming distributed multi-robot systems. In Proc. The Seventh International Symposium on Artificial *Life and Robotics (AROB 7th ' 02)*. Beppu, Japan.

Sapaty, P., Sugisaka, M., 2002a. Universal distributed brain for mobile multi-robot systems. In book *Distributed Autonomous Robotic Systems*. H. Asama, T. Arai, T. Fukuda, and T. Hasegava (Eds.), Springer-Verlag Tokyo, SPIN: 10869189.

Sapaty, P., Sugisaka, M., 2003. Optimized space search by distributed robotic teams. In *Proc. Symposium Unmanned Systems 2003*. Baltimore Convention Center, USA.

Sapaty, P., Klimenko, V., Sugisaka, M., 2004. Dynamic air traffic management using distributed brain concept. *Mathematical Machines and Systems*. ISSN:1028-9763, No.1.

Sapaty, P., Kawamura, K., Sugisaka, M., Finkelstein, R., 2004. Towards fully distributed cognitive systems. In *Proc. The Ninth International Symposium on Artificial Life and Robotics (AROB 9th ' 04)*. Beppu, Japan.

Sapaty, P. S., 2004. *Ruling Distributed Dynamic Worlds*, to be publ. by John Wiley & Sons, New York.