Article

Emergent Intelligence in Smart Ecosystems: Conflicts Resolution by Reaching Consensus in Resource Management

George Rzevski [1], Petr Skobelev [2,*] and Alexey Zhilyaev [3]

[1] Complexity and Design Research Group, The Open University, Milton Keynes MK7 6AA, UK; george@rzevskiresearch.uk
[2] Samara Federal Center of Russian Academy of Science, Studenchesky Str., 3A, 443001 Samara, Russia
[3] Department of Electronic Systems, Information Technology Faculty, Samara State Technical University, Molodogvardeyskaya Str. 244, 443100 Samara, Russia; zhilyaev.alexey@gmail.com
* Correspondence: petr.skobelev@gmail.com; Tel.: +7-902-372-3202

Abstract: A new emergent intelligence approach to the design of smart ecosystems, based on the complexity science principles, is introduced and discussed. The smart ecosystem for resource management is defined as a system of autonomous decision-making multi-agent systems capable to allocate resources, plan orders for resources, and to optimize, coordinate, monitor, and control the execution of plans in real time. The emergent intelligence enables software agents to collectively resolve conflicts arising in resource management decisions by reaching a consensus through a process of detecting conflicts and negotiation for finding trade-offs. The key feature of the proposed approach is the ontological model of the enterprise and the method of collective decision-making by software agents that compete or cooperate with each other on the virtual market of the digital ecosystem. Emergent intelligent systems do not require extensive training using a large quantity of data, like conventional artificial intelligence/machine learning systems. The developed model, method, and tool were applied for managing the resources of a factory workshop, a group of small satellites, and some other applications. A comparison of the developed and traditional tools is given. The new metric for measuring the adaptability of emergent intelligence is introduced. The performance of the new model and method are validated by constructing and evaluating large-scale resource management solutions for commercial clients. As demonstrated, the essential benefit is the high adaptability and efficiency of the resource management systems when operating under complex and dynamic market conditions.

Keywords: emergent intelligence; smart ecosystems; decision making; autonomous systems; complex adaptive systems; self-organization; coevolution; non-determinism; adaptability; ontology; multi-agent technology; real time

MSC: 68T20; 68T30

check for updates

Citation: Rzevski, G.; Skobelev, P.; Zhilyaev, A. Emergent Intelligence in Smart Ecosystems: Conflicts Resolution by Reaching Consensus in Resource Management. *Mathematics* **2022**, *10*, 1923. https://doi.org/10.3390/math10111923

Academic Editors: Jüri Majak and Ripon Kumar Chakrabortty

Received: 5 May 2022
Accepted: 1 June 2022
Published: 3 June 2022

1. Introduction

Natural ecosystems, such as forests, grasslands, deserts, rivers, and oceans, are *complex* and, therefore, have a remarkable ability to self-organize in order to adapt to external changes and to evolve with their environments, enabling some of them to sustain existence for many millions of years.

In contrast, artificial (man-made) systems are, as a rule, mechanical, algorithmic, and deterministic. By one definition, system engineering is the "design, building, and use of engines, machines, and structures" [1].

Even businesses are usually designed as rigid hierarchical corporations with top-down control and bottom-up reporting. Such hierarchical management structures were very effective under the conditions of stable markets with predictable supply and demand that prevailed during the last two centuries.

The 21st century has brought about a radical change. As billions of individuals and millions of businesses joined online trading, they triggered a steep increase in the *complexity* of the Internet-based global market to such a level that the market became volatile and supply and demand are now unpredictable. The successful performance of many corporations based on the rigid advanced planning of the resources is now seriously affected by the sharp increase of *disruptive events*, such as the cancellation or modification of orders, failures of overstretched resources, and electronic fraud, cyberattacks, or war conflicts [2,3].

It is quite clear that adaptable and sustainable businesses and administrations, just like natural ecosystems, would be much more effective under new complex market conditions. The proposed bio-inspired approach is particularly aimed at solving resource management problems caused by the recent exponential increase in market complexity.

From the authors' point of view, their method and tool of Emergent Intelligence for forming Smart Ecosystems are defining the "new mathematics" for the real time economy of 21-st century. This new mathematics will help to make a very important step from the model of classical centralized planning and optimization (which is focused on satisfying interests of the center only, not of its parts) and models of game theory (which is focused on fights) to the new model of identifying conflicts, finding balances of interests and reaching consensus among decision makers. Consensus is defined here as a "competitive equilibrium" in the collective decision making of agents, representing human, physical, or abstract entities, able to compete and cooperate in the virtual market of self-organized multi-agent systems and "systems of such systems", which form the new model of bio-inspired smart digital ecosystems.

This new approach shows the value and benefits in many practical applications where classical mathematical methods and tools (linear, dynamic or constraint programming, heuristics, etc.) do not provide feasible solutions or are not efficient. The efficiency of the new method and tool of Emergent Intelligence for solving complex problems, based on the principles of complexity science and self-organization of agents, is demonstrated in this paper using examples of scheduling and optimization applications, but it could also be applied to much wider domains, such as design and construction, text understanding, clustering, pattern recognition, and many others.

This paper is organized as follows. Section 2 is focused on real examples of complex resource management problems that illustrate the complexity and dynamics of decision-making processes. Section 3 will introduce Emergent Intelligence in smart ecosystems (systems of smart ERP solutions) as a new approach for solving the discussed problem. The design principles and logical architecture of a smart ecosystem are proposed in Section 4. The proposed emergent intelligence models and methods are specified in Section 5, including the ontological model of enterprise and multi-agent method of solving conflicts and finding trade-offs in the scheduling and optimization of resources. Section 6 presents an experimental study of the new approach and technology for the evaluation of the quality of results and system performance. The new metric for measuring the adaptability of emergence intelligence is introduced in Section 7. A discussion and evaluation of results are presented in Section 8. Section 9 gives conclusions and outlines future developments.

2. New Problems in Resource Management

In recent decades, a number of new methods and tools for enterprise resource management (ERP) solutions were developed, including various methods of combinatorial search, heuristics and meta-heuristics, local search, tabu-search, simulated annealing, ant and swarm optimization, genetic algorithms, and stochastic methods, etc. [4].

According to our previous study of existing ERP solutions [5,6], the developed solutions work relatively well in centralized and stable environments where orders are given in advance and resources do not change during the execution of plans. The key part of ERP solutions is planning and optimization, which is mostly work in batch mode and not in real time. However, traditional ERP solutions also ignore individual criteria, preferences, and constraints of many parties involved (clients, employees, partners, suppliers, etc.). These solutions require a step forward to the real-time networking, monitoring, and controlling of resources, to close the feedback loop in adaptive resource management.

Growing market complexity, uncertainty, and the dynamics of demand and supply form new requirements for resource management. Typical examples of complex and dynamic resource management applications [5] are presented in Table 1. The unpredictability and frequency of changes in demand and supply are such that current ERP systems, based on batch-mode optimizers, cannot be effectively used to solve the current complex problems of resource management with acceptable precision and within the available time frames.

Table 1. Examples of complex and dynamic resource management applications.

Types of Resource Management Problems	Examples of Applications	Complexity and Dynamics
Management of transportation resources	National size fleet of 1000 trucks and drivers, 3500 source and destination points, 300 orders per day.	Maintenance and fueling of trucks, compatibility of cargos, driver shift regulations, loading and unloading rules. Unpredictable arrival of orders, delays in deliveries, traffic jams, road incidents.
Management of supply chains	National size supply chain of drinks: 5 factories, 300 storages and cross-docs, 1000 shops.	Real time scheduling of products for 10 days horizon, coordination required between production and transportation, various transportation channels. Daily changes of forecast because of weather, sport events and other factors.
Mobile resource management	Regional taxi, food delivery services, technicians for gas, electricity and water supply.	Dependency of orders, pre-history, competencies of workers. Unpredictability of orders, delays in operations, unavailability of resources.
Factory management	Federal Air jet factory includes 20 workshops, each requires about 300 units of equipment and 150 workers.	Variety of orders, deep product break down structure, tasks interdependencies in technological processes, productivity and competencies of workers. Unexpected orders, breakdown of equipment, delays in operations.
Management of R&D projects	Airspace enterprises or regional ministry of economics: 100–300 projects, 1500 employees involved, require 2–3 years to implement.	Variety of projects in one pool of shared resources, multi-objective tasks, interdependency of tasks, balanced budget, competencies of employees. New unexpected projects, new unforeseen tasks, delays in operations, nonavailability of resources with required competencies.
Management of railways	Regional railways include 1000 km of two-way lines, 50 stations, 700 trains.	Different types of trains (high speed, passenger, cargo), topology of stations and lines, interdependencies of schedules. Unpredictable delays and breakdowns of trains and lines, geographical distribution.

In situations when traditional ERP solutions do not work perfectly, companies usually resort to manual scheduling by bringing in more managers, and as a result corrections to disruptions are delayed, a larger number of products are kept in storage, many orders are lost, quality and efficiency of services are reduced, and, as a consequence, prices for clients have to be increased.

As a result of such constraints and the limitation of existing ERP solutions, the competitive advantages of a business could be lost.

3. Emergent Intelligence and Smart Ecosystems

In this paper, the smart ecosystem for resource management is defined as a system of autonomous decision-making, multi-agent ERP systems (smart ERP systems) capable of allocating resources, planning of orders and optimization of resources, coordinating decisions, and monitoring and controlling results, in real time.

Let's introduce the emergent intelligence (EI) of a smart ecosystem ("system of smart ERP systems") as the collective intelligence of a group of partially autonomous software agents engaged in intense interaction among themselves and with their environment and which is not identifiable as a property of any individual constituent agent.

The EI enables software agents, which can work as a representatives of smart ERP systems or inside such smart ERP systems, to collectively resolve conflicts arising in resource management decisions by reaching a consensus through a process of detecting conflicts and negotiation for finding trade-offs at all levels of an enterprise and between these levels.

Smart ecosystems that exhibit EI need to be designed as a complex adaptive software system, in which a multi-level network of agents exchange digital messages with each other and with their environment, negotiating the best possible consensus-based solution to a given problem under the everchanging requirements and within tight time constraints. Under such conditions, the digital ecosystem must be time-sensitive and non-deterministic on all levels, in which the same inputs, as a rule, do not produce the same outputs; decisions depend on the time when they are made. In contrast, traditional ERP software is usually deterministic, where the same inputs always produce the same outputs.

EI is a fundamental property of all natural ecosystems which are sufficiently *complex* (due to high connectivity and partial autonomy of their participants engaged in competition or cooperation with each other) to exhibit emergent intelligence, adapt to environmental changes, and to coevolve with the environment [7].

The proposed approach for building smart ecosystems exhibiting the same features as natural ones, including EI, supports the autonomous management of resources under conditions of currently prevailing complexity.

The key novelty is that the new EI approach supports a process of coordinated collective decision making in self-organized systems capable of resolving conflicts between various demands and resources for reaching consensus among competing and cooperating digital agents. The consensus is defined as a solution to a problem in which all affected agents, not only neighbors, agree with each other. Participating software agents are fully or partially autonomous, have their own individual goals, and can discover and solve conflicts by negotiating trade-offs. The result of the applications of EI solutions to business is an increase in value generated by the chain of collectively taken decisions of agents when reaching consensus.

The key advantage of smart ecosystems, when used as models of business processes, especially of resource management, is that they can replicate in a digital space almost all real-life activities, without simplifying the *diversity* present in the real world (as it is necessary to do when using mathematical modeling) and, in addition, smart ecosystems *coevolve* with the business processes that they model; they can act as digital twins.

The autonomy of smart ecosystems is considered as the result of the convergence of many modern information technologies, such as the Internet of Things [8], cyber-physical systems [9,10], multi-agent technologies [11–13], smart robotics [14], model-driven simulations [15,16], ontology- and knowledge-based decision making [17,18], smart, cognitive and autonomous digital twins [19,20].

The modern trend in developing digital twins is focused on smart and cognitive features for better and faster adaptation [21–23]. EI of digital twins in smart ecosystems will play an important role in the future. For example, in [24], an EI approach is contrasted to multi-agent technology as a developed meta-algorithm and applied to solve a variety of resource allocation and job shop scheduling problems. In [25], EI is interpreted as the collective intelligence of a large swarm of identical or diverse robots. In [26], EI is applied to routing packages through mobile Wi-Fi networks. In [27,28], EI is applied for distributed stochastic optimization and managing a swarm of mobile robots. In the case of optimization, it is distributed sequential subspace optimization where coordination is considered among neighbors.

The authors discovered how complexity creates EI when they experimented with a complex adaptive software system for managing a rather difficult transportation problem [29]. Software agents, when confronted with an unusually complex scheduling problem, autonomously (without being instructed) and collectively decided to change the schedule in an unexpected manner, by a long chain of coordinated decisions, which led to the resolution of a critical issue.

In this case, the behavior of agents was similar to a chemical *self-catalytical process*, as explained by acknowledged fathers of complexity science, Prigogine and Kaufman [30–32].

4. Logical Architecture of a Smart Ecosystem

The smart ecosystem consists of a digital platform and autonomous decision-making systems, represented by digital agents, able to take and revise decisions for resolving conflicts. Each autonomous decision-making system is a goal-driven, knowledge-based, multi-agent system, which is able to analyze problems, plan their activity, and produce results with the use of available resources.

The main subsystems of a smart ecosystem include: real world (RW), virtual world (VW), knowledge base (KB), consisting of ontology and data, and EI decision making dashboard (EID) displaying the value created for the business, i.e., the profit.

Figure 1 shows the relations between RW and VW in smart ecosystems.

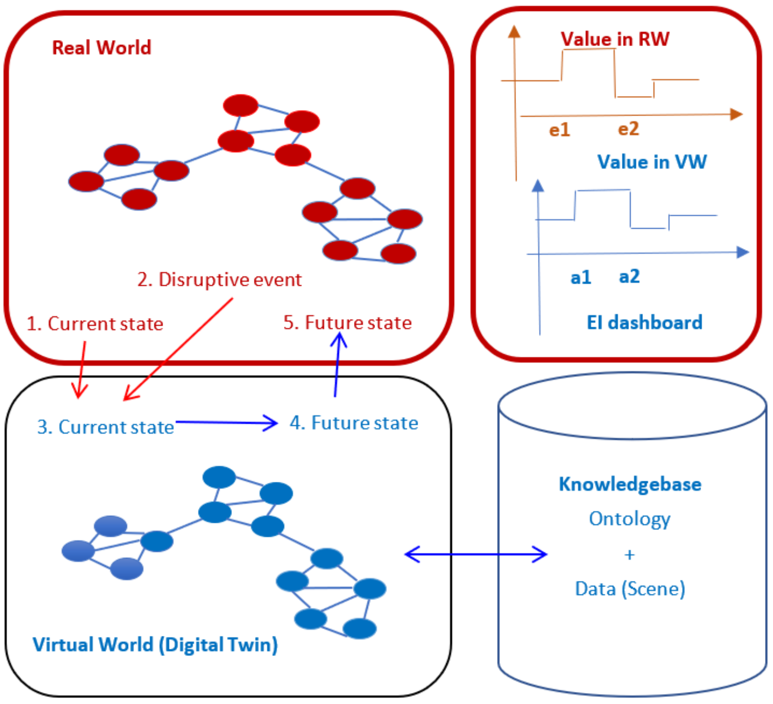


Figure 1. The relations between real world and digital world of digital ecosystem.

The knowledge base contains domain knowledge required to activate the ecosystem. It consists of two distinctive parts: ontology, which contains conceptual knowledge organized as a semantic network, and data (scene), describing the context of the situation. The scene is, in fact, the ontological description of the current state of the real world and the current schedule of all orders and resources.

The ontology can be updated whenever rules, regulations, or ecosystem policies are changed by authorized operatives without interrupting the ecosystem operation.

The collective decision-making of agents in smart ecosystems maximizes value in the case of positive events (new order arrives) or minimizes losses in the case of adverse events (resource is unavailable).

RW consists of human decision-makers, market demands (orders), and available resources. However, because human decision-makers cannot always achieve the required combination of decision-making speed and precision under current conditions of market complexity, the actual operational decision-making is delegated to VW.

VW is a complex adaptive software system consisting primarily of interacting demand agents representing the actual demands and resource agents representing the actual resources, designed to make resource allocation decisions instead of managers, being the real-world decision makers.

Digital agents allocate resources to demands through negotiation among themselves, consulting ontology and, if necessary, consulting human decision-makers.

The process flow is as follows. Let's assume that VW is in the initial state (1). When a disruptive event, e1, occurs in RW (2), agents in the current state of VW (3) rapidly detect the event, identify parts of the real world that could be affected, reschedule affected parts without disrupting the operation of the unaffected parts, and generate a new state of VW (4). Then, VW sends an instruction, a1, to the real world, to redeploy available resources to neutralize the disruption (5).

Arrows in Figure 1 mean that the current state in RW is translated into the current state of VW and that a disruptive event changes this state. These changes trigger the adaptive re-scheduling of resources with the objective to achieve and keep the value as high as possible. As a result, the system passes to the next state. It is a transition to the new attractor, which is considered as "competitive equilibrium" between demands and resources. The new finite state is sent back to the appropriate human resources of the enterprise in RW. The knowledge base is consulted before working out the solution.

All this is done in real-time e_1->a_1, e_2->a_2, ... , depending on the system performance and interoperability of services. The system, by processing events, keeps VW and RW synchronized, which enables VW to act as a digital twin of RW.

The essential advantages of the complex adaptive system, residing in VW, over conventional software are (a) the power of EI which enables the rapid assessment of system vulnerability to a disruptive event, followed by the elimination of the negative effects of the disruption by rescheduling only affected orders and resources without disturbing unaffected parts of the system, and (b) the speed of reacting to disruptive events, which is achieved by replacing the computational search for a solution by communication, i.e., the exchange of messages between agents.

Let us look now at how the system resolves conflicts. The new order agent locates conflicting orders for a specific resource from the records of all "order-resource" relations recorded in the scene. Using these relations, the order agent finds other orders which can be affected by the reallocation of the affected resource, selects the best reallocation, and sends the suggestion to relevant order agents, initiating conflict resolution negotiations. In the same way, agents use other ontology relations, such as "be part of" or "next task", stored in the scene for solving conflicts. The use of semantic relations also helps agents to reduce combinatorial search duration by analyzing the semantics of the problem domain. The use of ontology (semantics) and agent communications (the exchange of messages between agents) makes smart ecosystems adaptive, flexible, and efficient in real time resource management.

To the best of our knowledge, no other ERP systems have these features.

Authorized users can access and update knowledge base directly. Once the knowledgebase is modified, the changes are immediately transmitted to VW. For example, if a user adds a new requirement for one "task" class (e.g., the class "task" now requires workers with new competencies), then all agents of instances of this task class are immediately informed of the new requirement. As a result, task class agents will cancel existing relations and start scheduling from the beginning following with new requirements.

An advanced ontology design is in the pipeline, enabling ontology evolution by autonomously identifying and discarding the out-of-date knowledge elements and requesting from the users the updates.

The EI dashboard shows how the occurrences of new RW events trigger the rescheduling of resources in VW, display collective agent decisions triggered by every event, and depict RW value created by the agent negotiations, after the instruction for resource redeployment are conveyed from VW to RW.

5. Emergent Intelligence Models and Methods of Collective Decision Making

5.1. Ontology and Enterprise Knowledgebase

An ontology is defined as

$$O = (C,R,F), \tag{1}$$

where C is a set of concepts, R a set of relations, and F a set of semantic functions, providing access to concepts and relations. An ontology for allocating resources to demands contains two parts. The first part is a basic ontology, *Ob*. The basic ontology is formed by basic concepts, which are relevant for the resource management domain, in general, and not specified for any problem domain, such as the manufacturing of aircraft or satellites (see Table 2). A domain ontology, *Od*, contains domain-specific concepts, which are specific to each problem-domain application and extend basic concepts and relations:

$$Od \supseteq Ob \tag{2}$$

Table 2. Basic ontology objects for the allocation of resources to demands.

Order	**Specifies the Product, Quantity of Products, and the Period of the Order Fulfilling**
Product	The product can be consumed or produced
Task	Specifies required resources and the actions
Resource	Physical, human, and financial resources required

The extension of the basic concepts and relations in domain ontology helps to formalize enterprise specifics.

Concepts of the ontology, described in Table 2, can be formally specified as

$$\mathrm{C}_{plan} = \{Order,\ Product,\ Task,\ Resource\}. \tag{3}$$

Each order creates a product connected to an appropriate task:

$$\forall_x \exists_y\ (Order(x) \to Product(y)\ \wedge\ create(x,y)). \tag{4}$$

The ontology considers two types of products, which are "produced products" by business process or by task and "utilized products". The relation between tasks and produced or utilized types of products is given by:

$$\begin{array}{c} \forall_x \exists_y\ (ProducedProduct(x) \to Product(x) \wedge\ Task(y) \wedge produce(y,x)), \\ \forall_x \exists_y\ (ConsumedProduct(x) \to Product(x) \wedge\ Task(y) \wedge consume(y,x)). \end{array} \tag{5}$$

The set of tasks is partitioned into the subsets of atomic tasks and group tasks.

One of the most important classes of relations between tasks is "to-be part of" a business or technological process. Another key relation between tasks is "next-previous (is followed)". Using these relations, one can find all tasks which are required for executing the business or technological process and analyze previous or following tasks with the requirements for detecting and solving conflicts. This is extremely important for any smart

ERP when a new, unpredictable, and disruptive event is occurring and there is an urgent need to change the sequence of tasks:

$$\begin{gathered}\forall_{x,y}\ (partof(x,y) \rightarrow Task(x) \wedge\ Task(y)),\\ \forall_{x,y}\ (follow(x,y) \rightarrow Task(x) \wedge\ Task(y)),\\ \forall_{x}\exists_{y}\ (GroupTask(x) \leftrightarrow Task(x) \wedge\ Task(y) \wedge partof(y,x)),\\ \forall_{x}\ (AtomicTask(x) \leftrightarrow GroupTask(x)).\end{gathered} \tag{6}$$

The basic classes of tasks are described in Table 3, and resources in Table 4.

Table 3. Classes of basic tasks.

Class of Task	Functionality
Elementary task 1	The task must be completed within a specified time (fixed task duration)
Elementary task 2	Task duration depends on resources and/or product volume (fixed work volume)
Elementary task 3	The task must be accomplished in a correct sequence (hammock)
Composite task	Task duration equals the sum of elementary sub-tasks duration

Table 4. Classes of basic resources.

Class of Resource	Functionality
Consumable resource	The resource is required for task fulfilment and will be consumed in the process of the task fulfilment
Reusable resource	The resource is required for task fulfilment but will recover after the task fulfilment

The relation "require" describes the type of resources required for the fulfillment of a task. The concept "resource requirement" denotes a specification of types of resources required to fulfill a task.

$$\forall_{x,y}(require(x,y) \rightarrow Task(x) \wedge (ResourceRequirenment(y)\ \vee\ Resource(y))). \tag{7}$$

Products may need to be "stored",

$$\forall_{x,y}\ (stored(x,y)\ \rightarrow\ Product(x)\ \wedge\ ReusableResource(y)). \tag{8}$$

The set of basic relations can be formally described as

$$R_{plan} = \{create,\ consume,\ produce,\ partof,\ follow,\ require,\ stored\}. \tag{9}$$

Let us consider the interpretations (semantics) of concepts and relations, *F*. Functions *F* provide access to ontology and navigation through the semantic network in such a way that new concepts and relations in the domain ontology *Od* can extend the basic ontology *Ob* for customizing the decision-making logic of a Smart ERP solution. Examples for the manufacturing domain include: "products" in basic ontology are extended as "components", "assembly elements", or "final products" in the domain ontology. "Tasks" are extended as "processes" and "operations", and resources = as "machines", "equipment", or "employees":

$$\begin{gathered}\forall_{x}\ (Product(x) \rightarrow Component(x)\ \vee AssemblyElement(x)\ \vee\ FinalProduct(x)),\\ \forall_{x}\ (Task(x) \rightarrow Process(x) \vee\ Operation(x)),\\ \forall_{x}\ (Resource(x) \rightarrow Equipment(x) \vee\ Tool(x)\ \vee\ Employee(x)).\end{gathered} \tag{10}$$

Enterprise ontology M is defined as an extension of domain ontology by a set of instances I, as follows:

$$M = \{O_{domain}, I\}, \tag{11}$$

where I is considered a list of employees with identification numbers or as a list of equipment units with inventory numbers.

Enterprise scene S is defined as an instantiation of M which mirrors the state of the enterprise and provides values of attributes of all instances at time t:

$$S = M(t) \tag{12}$$

The enterprise scene is stored in a database which is used for navigation via a semantic network of instances of concepts and relations. For example, the instance of "Task" concept can help to find relations with previous and next tasks, required resources, input and output products, etc. This data structure aims to reduce the combinatorial search in collective decision making. The method enables building an ontological model of any domain-specific enterprise precisely depicting the current state of the enterprise order-resource schedule for any given horizon.

Figure 2 illustrates the basic and the domain ontology, respectively, for a manufacturing plant. The main element of the ontology is a "task" connecting all other elements. If a manufacturing plant consists of several workshops, the same domain ontology is applicable to all constituent workshops.

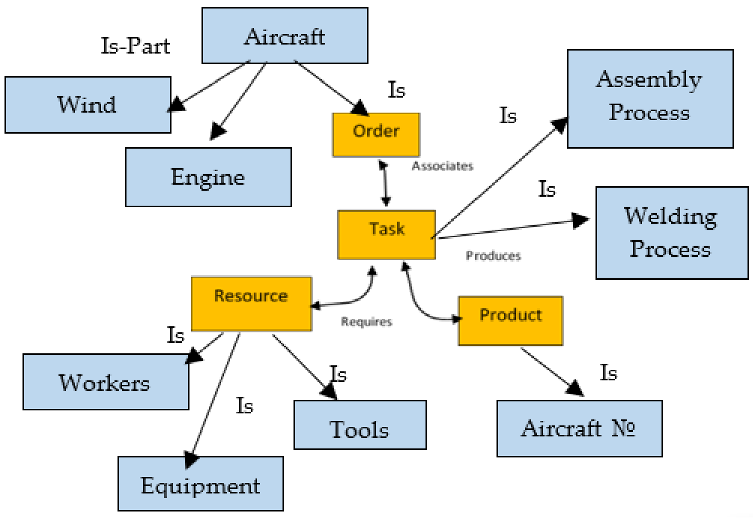


Figure 2. The basic (in yellow) and domain ontology for manufacturing (in blue).

Figure 2 shows that the basic ontology covers the management domain, in general, and it is not specialized for creating specific objects, such as aircrafts, satellites, etc. On the other hand, the domain ontology is industry-specific, and it extends concepts and relations of basic ontology to the enterprise domain, e.g., for manufacturing of aircrafts or satellites.

The objective of this formalization of the problem domain is to create an ontological model of enterprise, which gives the specification of classes of orders, resources, technologies, products, etc. This model can be loaded in the generalized and unified multi-agent system to provide customization of the solution for the business domain of the enterprise.

5.2. Virtual World

VW is triggered by a disruptive event from RW, either from the enterprise itself or its environment. The objective of Smart ERP is to minimize the disruption's negative consequences and increase value in case of positive events such as new order arrival. Each event triggers the wave of changes in the demand-resource schedule. The wave starts from the affected agent and ends by new "competitive equilibrium", which is formed by decision making and negotiation of agents to resolve conflicts and reach consensus.

The state of modelled enterprise is determined as a set of states of orders, products, processes and tasks, and resources:

$$\begin{gathered} S_{twin} = \{s_i\}, \\ s_i = \{model_i,\ plan_i, kpi_i\}, \quad 1, \ldots, n \end{gathered} \tag{13}$$

where S_{twin} is the state of a digital twin of the actual enterprise, represented by VW; $model_i$ is an ontological model of the object; $plan_i$ is an object schedule; kpi_i is a key performance index, n is the number of states. The main objective is that the state of the real enterprise S_{real}, and the state of its digital twin, S_{twin}, be always as close as possible:

$$D\left(S_{real}^{(k)},\ S_{twin}^{(k)}\right) \to 0. \tag{14}$$

where D is a function describing the difference between the actual enterprise order-resource schedule and the virtual world order-resource schedule. To achieve the same state, whenever a disruptive event, $Event^{(k)}$, occurs in the real enterprise, the operational schedule of the VW must adapt as quickly as possible to a new state:

$$S_{twin}^{(k+1)} = F\left(S_{twin}^{(k)}, Event^{(k)}\right), \tag{15}$$

where F, is the adaptation function. For every object in an ontology, s_i, a digital agent, a_i, is assigned to exhibit the object's behavior, as shown in Table 5.

Table 5. Main classes of agents.

Digital Agent Type	Goals and Preferences	Constraints
Order Agent	To be realized with minimum delay, c, and cost, p, $Y_i = w_1\left(1 - \frac{C}{C_{\text{кр}}}\right) + w_2\left(1 - \frac{p}{p_{\text{кр}}}\right)$	Time of delivery, volume, price
Task Agent (elementary, composite)	To be realized in match with required resource, before deadline ($\tau_i = finish_i - start_i$): $Y_i = \begin{cases} 1, \ \tau_i < \tau_{\text{опт}} \\ \frac{\tau_i - \tau_{\text{кр}}}{\tau_{\text{опт}} - \tau_{\text{кр}}}, \ otherwise \end{cases}$	Requirements of tasks, time of delivery, costs, links with previous and next tasks
Resource Agent	To maximize use of capacity $Y_i = \begin{cases} 0, \ u_i < u_{\text{кр}} \\ \frac{u_i - u_{\text{кр}}}{u_{\text{опт}} - u_{\text{кр}}}, \ otherwise \end{cases}$	Time of availability, matching rules for servicing
Product Agent	To provide storage To minimize idle time between production and consumption, e $Y_i = 1 - \frac{e_i}{e_{\text{кр}}}$	Storage size, time and cost of storage
Enterprise Agent	Objectives monitoring, load equalizing and coordination	Reaction time, length of modification chains

Note that if different types of orders or resources are considered, it is necessary to introduce new types of resource agents or to re-program the existing ones.

Objectives for every digital agent are defined using agent satisfaction, $Y_i(plan_i)$, which is a weighted sum of M elements belonging to kpi_i and calculated based on the current schedule, $plan_i$, related to the object agent as

$$Y_i = \sum_{j=1}^{M} w_{ij} y_{ij} \tag{16}$$

where y_{ij}, is an element of the satisfaction function defined by criterion $j = \overline{1, M}$, and w_{ij} is weighting coefficient $0 \leq w_{ij} \leq 1$ and $\sum_{j=1}^{M} w_{ij} = 1 \; \forall_i$.

The VW is organized as a virtual market where the order agents buy slots of time from resource agents, and vice versa. The conflict is detected in cases when several task agents demand the same slots of usage time from a resource agent, or a few resource agents contact the one order agent. The conflict with the new order in the schedule could be solved by open slot allocation, shifts, swaps, or drops of previously allocated orders.

Agents receive bonuses for an increase in satisfaction functions or penalties for underperforming and a decrease of satisfaction functions. For this purpose, each agent has a bonus (fines) function, $B_i(Y_i)$. The received or expected virtual money of bonuses can be used to compensate agents for the loss in satisfaction functions.

Various instances of basic agents may have different satisfaction and bonus/fines functions. The satisfaction function motivates the agent to achieve given objectives. Resource agents also have a cost function, related to the cost of their services. The selection and use of these functions depends upon the specifics of the problem domain. Their attributes could be defined by experts or managers of enterprises without re-programming agents. The only restriction is the computability of these functions regarding each state of each agent at any moment of time and their monotone that helps to reduce the size of the decision-making space of the Smart ERP solution.

The allocation of resources to orders is performed as follows.

In the beginning of the VW, S_{twin}, the number of instances of order agents, resource agents, task agents, and product agents are created and start interactions. Agent of new order, A_k, picks up from the knowledge base the business or technological process specifications and creates task agents connected by nesting or sequencing relations. A high-level task agent checks that relevant products and resources are available and ensure the task performance in the specified time. Each task agent starts with the analysis of required resources, comparing requirements and resource capabilities, resolving conflicts with previously scheduled orders. If the local optimization of agent actions is possible, the branch and bound method is applied. Identification of the set of conflicting orders can substantially reduce the number of solution options and speed up the search for the task fulfillment solution

$$\{a_i \mid i \neq k, \; plan'_k \cap \; plan_i \neq \varnothing\} \tag{17}$$

The conflicts are solved in the following way. Agent of new order selects the previously scheduled conflicting order which occupies preferred slot of time and sends request to find new allocation and sum of maximum compensation. The active agent tries to find the new position on the same resource or considers other options. If a new conflict is detected, the procedure repeats recursively. A resulting chain of modifications to the schedule produces losses suffered by agents who agreed to change their requirements to resolve a conflict, ΔB_i. A chain of modifications is deemed to be successful if the corresponding order agent is able compensate the losses of conflicting agents from gains earned by received bonuses, ΔB_k:

$$\Delta B_k \geq \sum_{i \neq k}^{n} \Delta B_i. \tag{18}$$

If this is the case, the schedule is accepted; if not, a new round of negotiations is performed. The order agent then identifies all products linked to it by the relation "produces" and informs their agents when they must be delivered to appropriate stores. The activity ends when a consensus is reached, i.e., when every agent a_k reaches a state in which no further adjustment of the schedule $plan'_k$ can improve their satisfaction function, ΔY_k, and consequently, increase their bonus function, $\Delta \mathrm{B}_k$, or the time available for negotiations runs out

$$\Delta B_k + \sum_{i \neq k}^{n} \Delta B_i < 0 \quad \forall_k. \tag{19}$$

Once a consensus is reached, the VW stops working and is switched to a standby mode, awaiting the next disruptive event or pro-activity stage for all agents.

The main KPIs are calculated by formulas presented as "Objectives/Goals" in Table 5. The procedures for these calculations are "built-in" (hardwired) in software agents. Input data, however, can differ for different agents. Agents read the ontology/knowledge base and start working on behalf of the objects that they represent, e.g., orders, tasks, products, resources, etc. Agents use the attributes of these objects and adjust their objectives accordingly. For example, the order agent will get type and name of the product that needs to be delivered and the required delivery time. The values of these attributes will be recalculated, and proper weights will be set in relevant formulae.

If the basic ontology requires changes, then typically the VW must be also redeveloped. In contrast, changes in domain ontology do not require the redevelopment of VW.

The main result of this section is to develop a generalized multi-agent tool for designing demand-resource networks of agents which can be customized for different Smart ERP applications. The objective functions of agents for manufacturing are given here as an example to show that not only classes of agents, but even each instance of agent may have its own individual objective function and individual bonus-penalty function. These functions are applied to measure increase in satisfaction and cost of this satisfaction (or vice versa) when agents solve detected conflicts and find trade-offs. These functions can be easily and flexibly modified for new domain applications and enterprise specifics without system reprogramming.

In the new approach, any enterprise entity, such as a person or equipment, can be modelled by an agent and becomes pro-active and self-optimizes with their own heuristics, but always in coordination with other agents.

The adaptability and efficiency of the new method is ensured by the use of protocols of conflict detection and resolution between orders and resources. In contrast to traditional combinatorial and heuristic methods, the interaction and computation starts among agents of conflicting orders and resources only. It significantly reduces computational complexity and provides the possibility to process event by event in a real-time mode.

If n is the number of orders, m is the average number of tasks in the technological process for each order and p is the number of resources which are in match with the order, then the number of interactions in the process of solving conflicts could be defined as

$$O(n \cdot m \cdot p \cdot (1 + conflict\_count)), \tag{20}$$

where the conflict count is the number of conflicts occurring during the allocation of an order, including recursive reallocations. In practical tasks, the value of p is significantly less than the number of resources, and the conflict count is limited by the compensation that the agent is ready to pay for new allocation. As a result, only neighbors are involved in conflict resolution.

The level and number of conflicts can be limited in practice.

In all cases, the more detailed and individual requirements orders have, the better and faster the results of the method are provided.

The fragment of the modified protocol of agent interaction and coordinated decision making for solving conflicts and reaching consensus is presented in Figure 3.

Agents of orders send requests to the agent of enterprise to allow improvements of their states in the schedule. The enterprise agent, in accordance with the selected strategy, sends approval for selected orders (1). If the order is already allocated in the schedule, it releases resources (2). Then, all orders are trying to find better resources (3). The order agent makes the proposal to all conflicting orders to find an alternative allocation and gives estimates of their costs, which could be compensated by this order. The procedure of finding new allocations (0) is recursive; the conflicting orders perform the same conflict resolution.

When an order receives information from conflicting orders, it takes the decision which reallocation to choose. If reallocation is not profitable, the order informs other agents that it cancels reallocations and asks the enterprise agent to allow a new search. It also stores information about the most expensive orders to avoid conflicts in the next round of interactions. If reallocation is profitable, it is accepted and committed to the current enterprise scene.

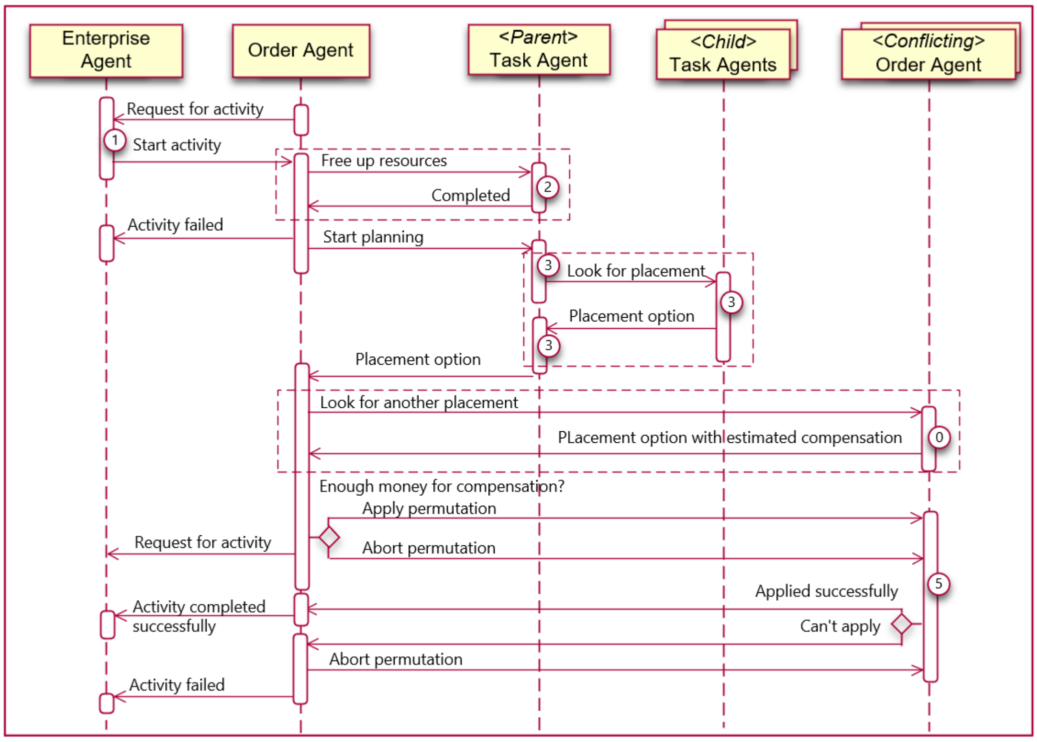


Figure 3. The modified protocol of agent interaction and coordinated decision making for solving conflicts and reaching consensus.

Another fragment of agent negotiations is given in Figure 4. The logic of the pre-matching of orders is encapsulated in the behavior of the demand sub-agent, which is generated by order agent. This agent send requests and selects only resource agents which are in full or partial match defined by specific rules from the ontology. In the next step, this agent is able to find the combination of required resources (4). The selected resources receive requests, release conflicting orders, and try to make a new allocation, to compare results with the previous state of the schedule.

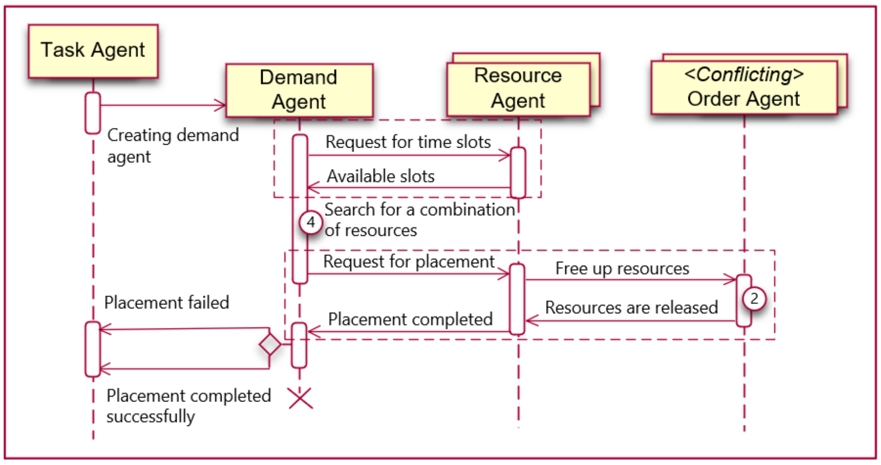


Figure 4. Fragment of negotiations of demand agent.

Let's consider an example of agent negotiations with the use of functions of satisfaction and bonus-penalties:

(1) Let's define the satisfaction function of agents of orders by a preferred time and the satisfaction function of agents of resources by a relation of working time to idle time. Bonus-penalty functions are introduced as linear functions with the domain of values [−100; 100] for orders and [0; 100] for resources.

(2) Let's imagine that, in initial state S_0, for resource R_1, orders O_1 and O_2 are already allocated. The order O_3 is coming (Figure 7).

(3) The agent O_3 starts looking for options. The most preferable position for this agent is t_1 (it associated with state S_1, see Table 7). To get this position, order agent O_3 sends message to agents O_1 и O_2 asking them to release this place on the resource schedule, find a new allocation, and send back the sum of costs for compensation.

(4) In this new situation, agent O_1 will agree to change his allocation and take position after agent O_3 in the case of compensation in 100 units (Table 6).

(5) The agent O_2 cannot find a new allocation (there is no time available on this resource for 3 orders execution) and demands compensation in 200 units.

(6) The agent R_1 reports that it can compensate 20 units only, which will be received by him because of a reduction of idle time (order O_3 requires more time than O_2).

(7) The order agent O_3 at the moment of time t_1 will get bonus 200 units but it will be not enough to compensate the sum of losses of other agents, which is equal 220 units. That is why this variant is rejected.

(8) The agent O_3 will consider the next option to be allocated at the moment t_2 (state S_2), and this requires solving the conflict with order O_2.

(9) The bonus for O_3 (160 units) will be enough to pay compensation to the conflicting agent (120 units). A decision will be taken, and all parties involved will be instructed to change their positions in the schedule (Figures 5 and 6).

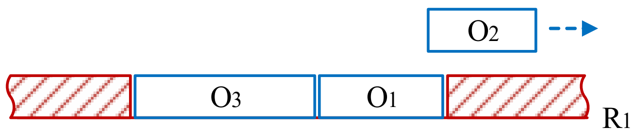


Figure 5. Reallocation of orders after negotiations.

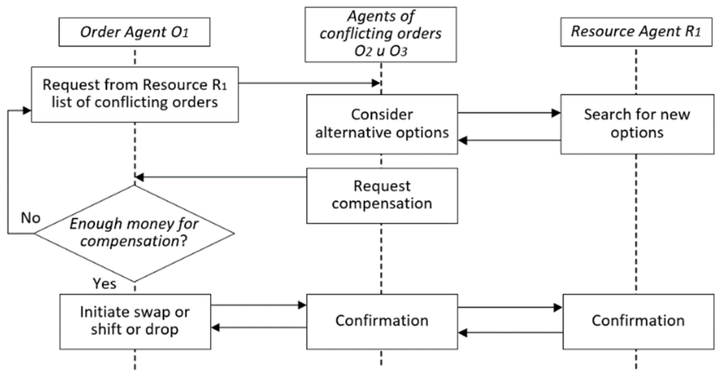

Order Agent O1
Agents of conflicting orders O2 и O3
Resource Agent R1
Request from Resource R1 list of conflicting orders
Consider alternative options
Search for new options
Request compensation
No
Enough money for compensation?
Yes
Initiate swap or shift or drop
Confirmation
Confirmation

Figure 6. A fragment of agent negotiation aimed at resolving conflicts.

Table 6. The change of agent budgets in transition between states.

Agent	$S_0 \rightarrow S_1$	$S_0 \rightarrow S_2$
O_1	−100	0
O_2	−140	−140
O_3	200	160
R_1	20	20

Table 7. The satisfaction and budget of agents in different states.

Agent\State	S_0		S_1		S_2	
	$\boldsymbol{Y}$	$\boldsymbol{B}$	$\boldsymbol{Y}$	$\boldsymbol{B}$	$\boldsymbol{Y}$	$\boldsymbol{B}$
O_1	1.0	100	0.5	0	1.0	100
O_2	0.7	40	0.0	−100	0.0	−100
O_3	0.0	−100	1.0	100	0.8	60
R_1	0.7	70	0.9	90	0.9	90

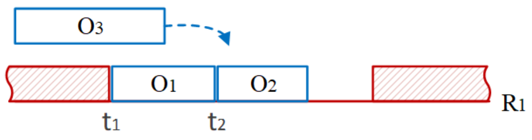


Figure 7. The initial state of orders and resources.

The resulting schedule is the outcome of collective decision making and negotiations of agents of orders and resources which take coordinated decisions and find trade-offs in the virtual market of the multi-agent system. In this sense, the schedule is self-organized by agents forming the Emergent Intelligence of the ERP solution.

5.3. Multi-Layer Digital Ecosystems

It is occasionally necessary to design a multilevel digital ecosystem to model the hierarchical structure of the real world.

In a multi-level virtual world, where each agent represents a lower-level digital ecosystem, agent negotiations may be horizontal and vertical. Horizontal negotiations include negotiations between multi-agent systems on one level, while vertical negotiations include those between multi-agent systems on a few levels. Nevertheless, any level may include intermediate VW, which contains agents-representatives of low-level multi-agent systems.

When VW models a traditional corporation, it may have many levels, e.g., holding enterprise, branch, unit, and workshop, and at each level, a fully self-contained digital ecosystem is allocated (Figure 8).

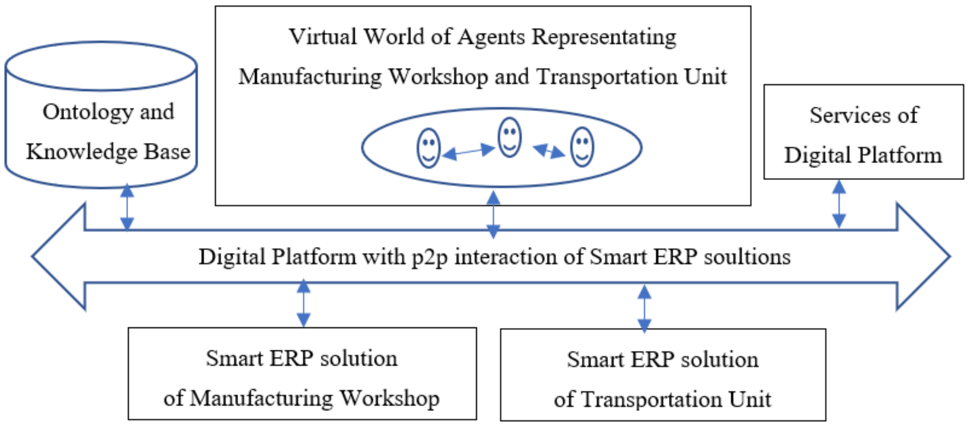
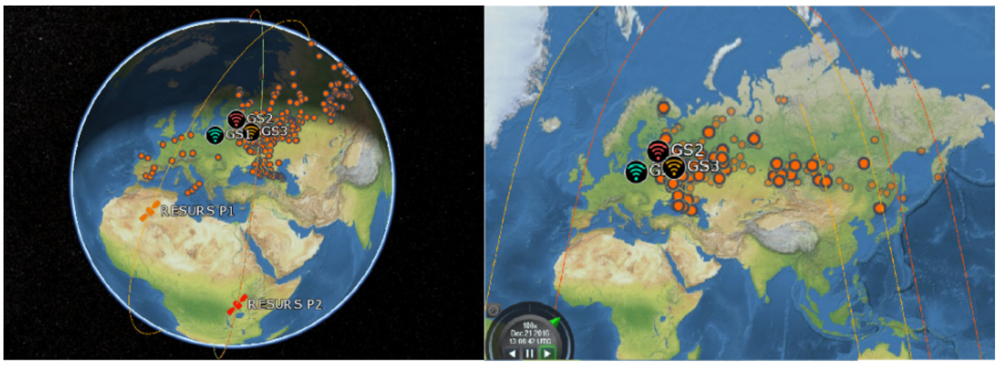

Virtual World of Agents Representating
Manufacturing Workshop and Transportation Unit
Ontology and
Knowledge Base
Services of
Digital Platform
Digital Platform with p2p interaction of Smart ERP soultions
Smart ERP solution
of Manufacturing Workshop
Smart ERP solution
of Transportation Unit

Figure 8. The fragment of smart manufacturing ecosystem which supports p2p interaction of smart ERP solutions for manufacturing and transportation.

Figure 9 shows a fragment of such a digital ecosystem that includes smart ERP solutions which are designed as ontology-driven multi-agent systems for factory manufacturing and transportation workshops.



(**a**) (**b**)

Figure 9. The number of orders for distant observation of Earth: (**a**) Geographical distribution of 3 satellites and 7 ground stations; (**b**) the initial set of orders for distant observation of Earth.

In this case, the smart ERP solution for manufacturing creates a manufacturing plan for a day, including demand for transportation. This production plan is then sent to a smart ERP solution of a transportation unit considered as a demand and triggers the scheduling of trucks, which are, in turn, sent to the smart ERP solution for manufacturing for consideration. These horizontal agent negotiations are carried in the VW of a factory containing agent representatives of both workshops.

Both smart ERP solutions continuously react to any proposed change in the capacity or the occurrence of disruptive events and renegotiate their schedules synchronically. As a result, smart ERP solutions coevolve in synergy.

Let us consider examples of p2p interaction between smart ERP solutions with coordinated collective decision making.

Disruptive event 1: Delay in manufacturing.

Smart ERP of the manufacturing workshop recognizes that a previously agreed plan is not valid because of the delay of supplying sub-components and informs Smart ERP of the transportation unit about this delay. Smart ERP of the transportation unit attempts to reschedule the previously allocated truck but finds that a truck of the required size is not available and has to schedule a larger truck. Reacting to this change, Smart ERP of the manufacturing workshop re-schedules the manufacturing process to produce more products and fill the larger truck.

Disruptive event 2: The scheduled truck breaks down.

Smart ERP of the transportation unit attempts to replace the failed truck but finds that only a small size truck is now available. Consequently, the smart ERP of the manufacturing workshop must return to the initial schedule.

The digital platform of VW is designed to supports the p2p interactions described above. It enables software agents (as representatives of Smart ERP of lower levels) to exchange messages with agents operating in a high-level digital ecosystem, which is engaged in the long-term planning, a unique feature that enables the synchronized allocation of resources at strategic and operational levels.

6. Experimental Study of the New Technology

New ontology-based, multi-agent models and methods for conflict resolution were applied to the scheduling of a group of 3 satellites of RESOURCE-P type for distant observation of Earth, operating in the network of 7 data transition stations.

The multi-agent Smart Satellites system was developed to provide real time adaptive scheduling of orders for satellites. The knowledge base specifies the types of orders, tasks, such as "focus camera", "take image", "save image", and "send image to ground station", satellites, and ground stations, channels of information, and image transmitting, etc. The agent of each satellite is able to compute the time zone when the required object and ground stations are visible, duration of camera focusing and imaging, resolution of image, size of image, and volume of available memory in the satellite. Order agent computes ballistics and determines ground stations, which are preferred for receiving images. Order agents are searching for satellites, and satellite agents are searching for orders, until "competitive equilibrium" of the virtual market is reached. The system automatically manages schedules for groups of satellites and ground stations for the horizon from 1.5 to 10 days.

The new method and system were compared with combinatorial depth-first search. The number of orders for distant observation of Earth was increased in iterations form 50 to 500 by steps of 50 orders (Figure 9).

Examples of screens of the developed framework are presented in Figure 10.

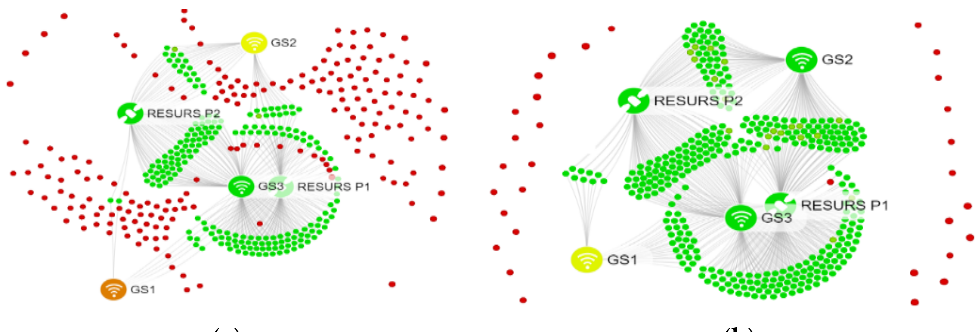


Figure 10. Examples of screens of the toolset: (**a**) initial state where many new orders (red circles) are not allocated to satellites (RESOURCE) and ground stations (GS); (**b**) the final state of consensus where only few red orders are still not scheduled.

The final allocation result shows that the network has insufficient resources for scheduling 500 orders, although most of the orders have been scheduled successfully, as represented by green circles. A much smaller number of red circles shows unscheduled orders.

An example of the schedule for group of the satellites is shown in Figure 11, demonstrating the number of scheduled orders and downloaded images and the level of satellite and orders satisfaction functions. The presented four diagrams provide details of the satisfaction function of orders and satellites, satisfaction function of orders, number of downloaded images, and number of orders for satellites accordingly. The satisfaction function of orders and satellites shows the impact of each negative event, e.g., when a satellite fails. The satisfaction function of orders is red in the beginning but then most orders get better satisfaction by resolving conflicts. The number of downloaded images and number of orders for satellites shows the allocation of orders per satellite and per ground station.

The results of experiments in terms of quality and computational time, and the comparison with the classical combinatorial method are shown in Figure 12.

Experiments show that the new approach provides high scheduling speed with a small reduction in the quality of results compared with the full-scale combinatorial search, which gives the global optimum but is applicable in batch computation only and therefore cannot be applied in real-time scheduling under conditions of market complexity.

In markets in which the interval between two consequent occurrences of disruptive events is short, the speed of searching for the best solution is more important than the ability to achieve the theoretical optimum. There is simply no time to wait for the traditional optimizers to complete computation; real-time scheduling is a must.

The new technology provides not only the full autonomy in processing new orders or other events, but also provides reasonable quality and real time adaptability of new order scheduling, as well as an opportunity to process more orders.

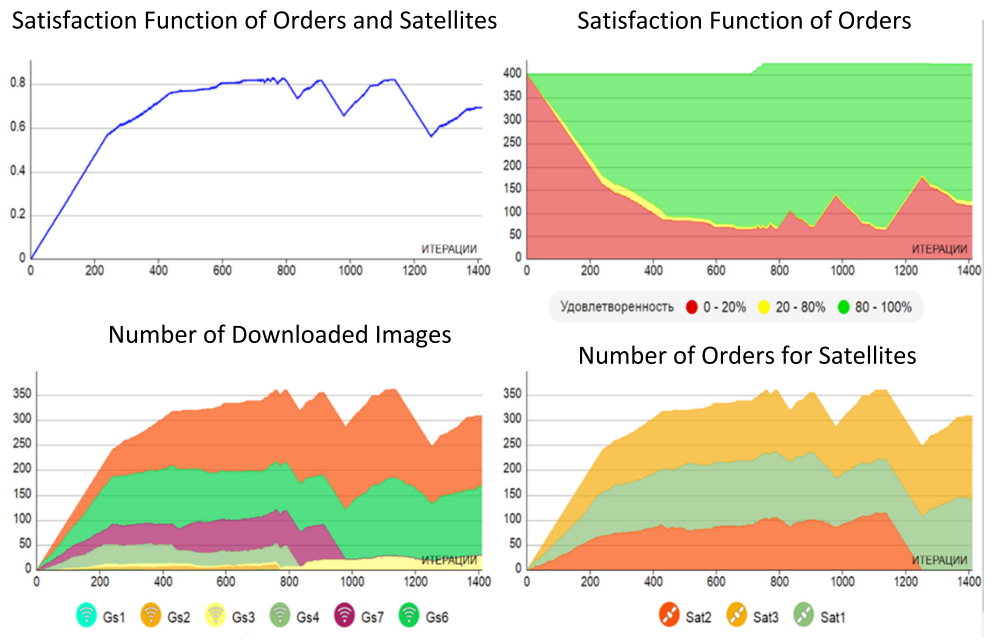


Figure 11. The example of resulting schedule of group of satellites.

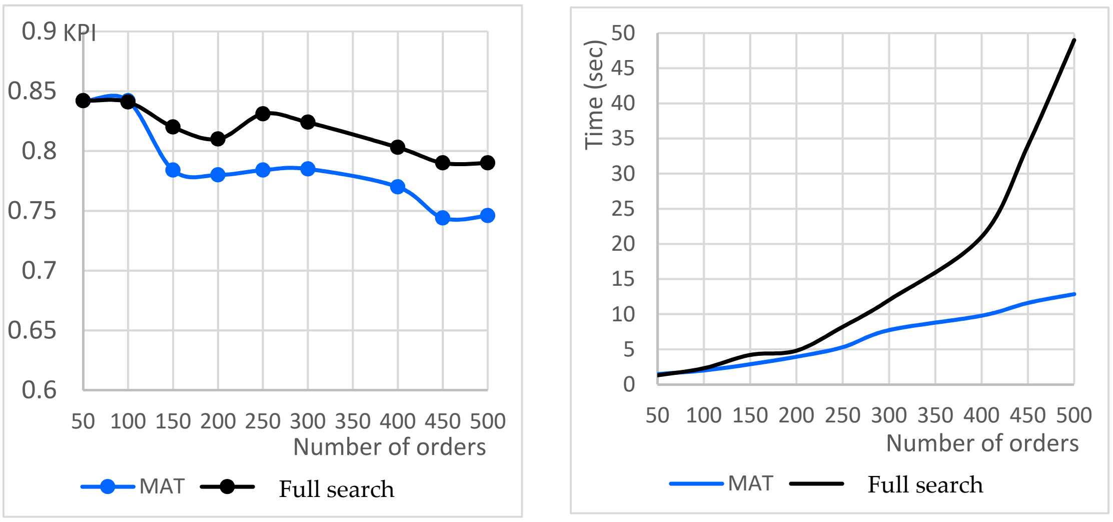


Figure 12. Quality and computational time achieved using the developed method (blue line) comparing with the full combinatorial search (black line).

7. Measuring the Effectiveness of EI in Reducing the Impact of Disruptive Events

EI reduces the negative consequences of unpredictable, disruptive events in complex environments by real-time rescheduling orders and resources affected by the disruption.

The functionality of EI provides an opportunity to introduce a new EI metric which helps to measure value of EI for business. In the case of negative events (such as “resource is unavailable”), EI is able to react in a way to minimize losses, and in the case of positive events (“new order has arrived”) to maximize profits.

Let us assume that the VW of a digital ecosystem is in the state S1(t1) and its satisfaction function has the value U1(t1) when a new disruptive event, Event(t1) (say, failure of a few resources), occurs (the number of resources reduces from N2 to N1). Previously agreed links between agents of orders and these resources are canceled, and order agents immediately begin attracting available resources and negotiating among themselves the resolution of newly emerging conflicts. As a result, the detection of the event immediately triggers the adaptive rescheduling of resources by EI and the VW will at time t2 reach a new state S(t2) in which, most likely, the value of the satisfaction function U(t) at time t2 will be less than it was at time t1 because not all orders can be satisfied to the previous value in the new state (lack of resources).

The adaptation process to new circumstances is illustrated in Figure 13.

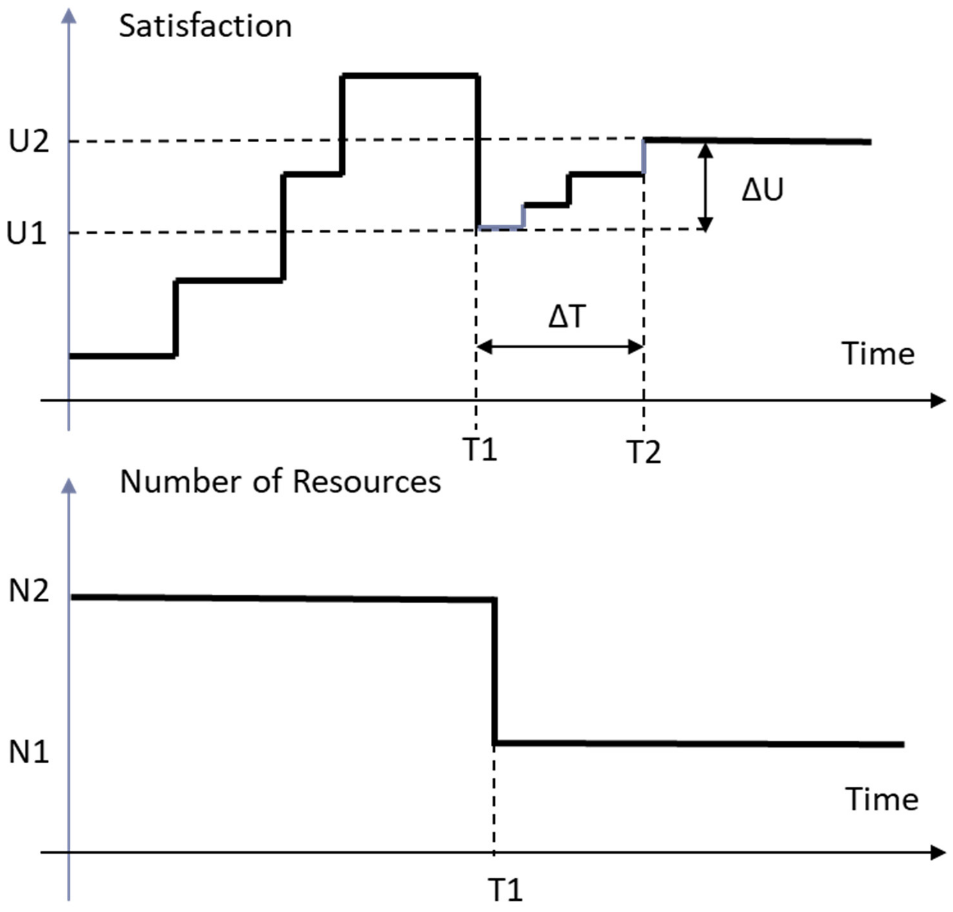


Figure 13. Adaptation by EI in case of an unpredictable, disruptive event.

Let's consider the satisfaction function for the system as a whole.

As the VW changes state, the new average value of the satisfaction function

$$u(t) = \frac{\sum_j u_j^{task}(t) + \sum_l u_l^{res}(t)}{M(t) + N(t)}, \tag{21}$$

where u_j^{task} is satisfaction function of the order j agent; u_l^{res} is satisfaction function of the resource l agent; $N(t)$ and $M(t)$ are the number of are order agents and resource agents, respectively.

Let us assume that the impact of the disruptive event reduced the value of satisfaction function to the value U_1 and that the system managed, due to its EI, to recover the value to U_2 in time $T = T_2 - T_1$. Then, we can postulate that the effectiveness of the EI of the VW is

$$\gamma = (u_2 - u_1)\cdot\frac{1}{T}. \tag{22}$$

The implication is that the higher EI of a digital ecosystem, the more successful the recovery after a disruptive event, and vice versa.

However, the behavior of digital ecosystems is not repeatable, and therefore the above measure is applicable specifically to a particular disruptive event in a particular context.

8. Discussion and Evaluation of Results

In contrast to deterministic systems, the complex systems are designed as self-organizing systems that form the solution of any resource management problem as a "competitive equilibrium" (consensus) of agents operating on the virtual market of ecosystems.

The proposed ontology-based collective decision-making of agents is driven by detecting and solving conflicts via negotiations and finding trade-offs. As found in experiments and applications of EI, the nature of the developed process of collective negotiations in smart ecosystems is similar to phenomena of autocatalytic processes discovered in organic liquids by I. Prigogine [30,32]. This phenomenon is considered as one of the fundamental factors in explaining the emergence of life in the universe, which is fueled by self-organization and natural selection, as postulated by S. Kaufman [31].

The authors consider that the emergence of life and the emergence of intelligence must be based on the same principles.

The features of artificial (designed) self-organized systems which demonstrate EI are different from the traditional software solutions, including:

- *Dependence on the past*: Results depend on the history of disruptive events and, consequently, the outcomes of agent negotiations are, in principle, unrepeatable.
- *Irreversibility*: Decisions are often irreversible. They cannot be always rolled back.
- *Butterfly effect*: Small changes may be amplified and cause significant changes in results.
- *Nonlinear effects*: Such effects as oscillations or catastrophes can propagate through the agent negotiations and cause unexpected chains of changes and delays in reaching a decision.
- *Loss of causality:* The system occasionally reacts to insignificant events by massive collective decision making.

These new features make smart ecosystems much more difficult to design, but also provide important benefits:

- *Broad scope of problem solving*: The new resource schedulers can cover various domains, including manufacturing, retail, financial services, administration, healthcare, and defense.
- *Practicality*: In cases where the time available for the search for a solution is very short (in some applications few seconds), the new technology cannot guarantee the global optimum but provides acceptable quality under circumstances. In cases where the time available for problem solving is not critical, it provide solutions equal to results obtained by linear programming [33,34].
- *Performance and Scalability*: The number of agents in a swarm and the number of swarms in a VW can be increased to several thousands of agents and more.
- *Adaptability*: The system reacts to a disruptive event with the speed appropriate to the frequency of the event occurrence.
- *Reliability:* Removing any part of the system, e.g., some part of the schedule, will only trigger the re-scheduling.
- *Self-improvement*: Agents are searching for performance improvements whenever not engaged in reacting to disruptive events.

As a result, the new EI technology provides acceptable (probably, the best possible), approximate solutions to complex NP-hard problems.

The EI technology was evaluated by solving a considerable number of large-scale commercial systems (Table 8), observing their behavior, and revising the initial conjecture, if necessary, as suggested by Popper in his seminal work on the epistemology of science, conjectures, and refutations [35].

Table 8. A selection of industrial applications of complex adaptive systems exhibiting emergent intelligence, which were used to evaluate EI technology.

Client/Type of Management Problem	Solution/Functionality	Measured Results
Rocket Space Corporation "Energeia": managing deliveries of 3500 cargos for 4 spaceships for the International Space Station.	Smart Aerospace: Complex adaptive real-time scheduler for delivery of cargo to the International Space Station.	Work time of the International Space Station and spaceships managers decreased up to 4–5 times. Simulation of scenarios provided data for risk assessment.
Managing supply chains for Lego (Chicago), Coca Cola (Germany), Gaspromneft and Siberian Coal Mining Company: real time scheduling of 1000 s products and resources.	Smart Supply Networks: Complex adaptive supply chain scheduler driven by real time events, including factories, storages and transportation channels	Results in Coca-Cola: Profit increased by 18%, order fulfillment increased by 7%, cost of transport reduced by 20%.
Addison Lee, London: real-time scheduling of 2000 taxis in the city center with up to 13,000 orders in peak-times.	Smart Taxi: Complex adaptive taxi scheduler—adaptively re-schedules new order up to 4 times before order confirmation for client and driver.	Idle runs reduced by 22.5%, fleet utilization increased by 5%, pick-up delays reduced 3 times, lost orders reduced by 2%.
TyazMach, Axion Holding, Airbus, Irkut, AviaAgregat, Kuznecov: managing workshops (150 workers) in daily operations.	Smart Factory: Complex adaptive production scheduler (EU Integrated Project "ARUM"): adaptation of personal schedules for workers in real time	Results in Axion Holding: Efficiency of workshop increased up to 10%, reducing 4 man-month of management (monthly).
Russian State Railways: Moscow-Saint—St. Petersburg region (50 stations/700 trains).	Smart Railway: Complex adaptive train scheduler: adaptive scheduling of stations and trains in reaction to disruptive events.	Delays reduced by 15% to 25%, speed of reacting to disruptive events increased 2 to 3 times, train speed increased by 3% to 5%.
Rocket Space Corporation Energia, Ministry of Economics, Samara Region.	Smart Projects: Complex adaptive project management system: adaptive re-scheduling of tasks of employees triggered by disruptive events and new projects.	Costs reduced by 5% to 10%, number of projects within the budget and deadline increased by 15%, transparency of projects delivered from annual plans to daily schedules of departments and individuals.
Prologics, Lorry, Monopoly, Trasko, Trans-Terminal: FTL/LTL logistics for fleet of up to 1000 trucks.	Smart Trucks: Complex adaptive road transport scheduler: adaptive rescheduling of orders to trucks and drivers.	Orders increased by 3% to 5%, delays reduced by 5%, utilization of resources improved by 5% to 10%.
Samara gas company, Volgograd water supply company, Far East service company.	Smart Field Service: Complex adaptive scheduler for field service technicians: adaptive re-scheduling of the allocation of tasks to workers in a team.	Reaction time to disruptive events decreased 5 to 7 times, productivity increased by 40% (7 orders a day increased to 12).
Instamart Moscow: on-demand food delivery from supermarkets to citizen.	Smart Food Delivery: Complex adaptive delivery scheduler—adaptive scheduling of couriers for food delivery.	Time for assembling the delivery reduced by 15%, delivery delays reduced by 22%.

The new method and the associated tool provide two very important features:

(a) the ability of the system to solve complex problems in different problem domains;
(b) to solve complex problems adaptively, reacting to disruptive events, in real time.

The new mathematics of this method is the mathematics of solving conflicts and finding trade-offs, balances of interests, and consensuses for the new networking economy instead of traditional centralized top-down planning and optimization or games theory.

Some of the above applications of EI resource management solutions are described in more detail in [36,37].

9. Conclusions and Future Developments

A new emergent intelligence approach to the design of smart ecosystems, based on the complexity science principles, is introduced and discussed.

The new method is based on the generalized and unified multi-agent world, designed for real-time planning, scheduling, and optimization organized as a collective decision making by software agents, which contains a dispatcher of agents, basic classes of agents, a messaging system, protocols of agent negotiations, etc.

The associated tool supports the self-organization of goal-driven demand-resource agents, which recognize and resolve conflicts by reaching consensus in a virtual market of the multi-agent system. The virtual market provides the possibility for agents to negotiate auction-like deals and find trade-offs, working in parallel, non-deterministically, and in an asynchronous mode. The demand-resource agents have conflicting objectives. For example, agents of orders have the objective to minimize delivery time and costs, agents of resources to maximize utilization, and agents of products to minimize idle time. The key idea of the new method is to find a balance of these interests and reach consensus by finding trade-offs measured by satisfaction functions. Hundreds and thousands of such agents, collectively taking decisions and continuously negotiating in a virtual market, form "emergent intelligence" based on the principles of complexity science.

This process of self-organization stops when a consensus of agents as a "competitive equilibrium" in a virtual market of the multi-agent system is reached.

The smart ecosystem for resource management is defined as a system of autonomous decision-making multi-agent systems (smart ERP systems) capable to allocate resources, plan orders for resources, and optimize, coordinate, monitor, and control the execution of plans in real time. The emergent intelligence enables software agents to collectively resolve conflicts arising in resource management decisions by reaching a consensus through a process of detecting conflicts and negotiation for finding trade-offs.

The key feature of the new approach is the ontological model of the enterprise and the method of collective decision-making by software agents that compete or cooperate with each other on the virtual market of the digital ecosystem. Emergent intelligent systems do not require extensive training using a large quantity of data, like conventional artificial intelligence/machine learning systems. The new method, and tool were applied to managing the resources of a factory workshop, group of small satellites, and some other applications. The comparison of developed and traditional tools shows the adaptability and advantages of developed methods and tools for solving complex resource management problems. The newly introduced metric helps to measure the adaptability of emergent intelligence solutions.

The performance of the new model and method are validated by constructing and evaluating large-scale resource management solutions for commercial clients.

As demonstrated, the essential benefit of the new approach is the high adaptability and efficiency of the resource management systems when operating under complex and dynamic market conditions.

Plans for future research of EI technology include:

The introduction of new measures of the EI effectiveness under various conditions, including the number of agents or links between agents that is affected by a disruptive event; the number of messages generated during conflict detection and related negotiations; time required for an agent to change its decision.

Adaptive ontology: The introduction of ontology agents with a role to monitor the use of the ontology and modify object classes, relations, or scripts to adapt the ontology to changes in demands.

Learning from experience: The introduction of additional agents to collect data and run machine learning algorithms for improving the ontology and system performance.

The development of an open-source EI platform for academic research and industry applications.

Future developments will also consider new application domains, including design and engineering, data mining and knowledge discovery, pattern recognition, healthcare, and agriculture, etc.

Author Contributions: Conceptualization, G.R.; methodology, P.S.; software, A.Z.; validation, P.S.; formal analysis, A.Z.; investigation, G.R.; resources, P.S.; data curation, A.Z.; writing—original draft preparation, G.R.; writing—review and editing, P.S.; visualization, A.Z.; supervision, G.R.; project administration, P.S.; funding acquisition, P.S. All authors have read and agreed to the published version of the manuscript.

Funding: The Russian team of research was funded by *Ministry of Science and Higher Education of the Russian Federation*, grant agreement number 075-03-2021-651/4.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Not available.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

RW	Real World
VW	Virtual World
Ob	Basic Ontology
Od	Domain Ontology
ERP	Enterprise Resource Planning
EI	Emergent Intelligence
KPI	Key Performance Indicator

References

1. International Council on Systems Engineering (INCOSE). Available online: https://www.incose.org/ (accessed on 10 March 2022).
2. Panetta, K. Gartner Top 10 Strategic Technology Trends for 2019. Available online: https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2019/ (accessed on 15 March 2022).
3. NATO Science & Technology Trends 2020–2040. Available online: https://www.nato.int/nato_static_fl2014/assets/pdf/2020/4/pdf/190422-ST_Tech_Trends_Report_2020-2040.pdf (accessed on 5 May 2022).
4. European Operational Research Conference. Available online: http://euro2018valencia.com/ (accessed on 15 March 2022).
5. Marik, V.; Gorodetsky, V.; Skobelev, P. Multi-Agent Technology for Industrial Applications: Barriers and Trends. In Proceedings of the 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2020), Toronto, ON, Canada, 11–14 October 2020; pp. 1980–1987.
6. Rzevski, G.; Skobelev, P. *Managing Complexity*, 1st ed.; WIT Press: London, UK; Boston, MA, USA, 2014; p. 216.
7. Ellis, E.C. Ecology in an Anthropogenic Biosphere. *Ecol. Monogr.* **2015**, *85*, 287–331. [CrossRef]
8. Alkhabbas, F.; Spalazzese, R.; Davidsson, P. An agent-based approach to realize emergent configurations in the internet of things. *Electronics* **2020**, *9*, 1347. [CrossRef]
9. Leitão, P.; Colombo, A.; Karnouskos, S. Industrial automation based on Cyber-Physical Systems technologies: Prototype implementations and challenges. *Comput. Ind.* **2016**, *81*, 11–25. [CrossRef]
10. Karnouskos, S.; Leitao, P.; Ribero, L.; Colombo, A. Industrial Agents as a Key Enabler for Realizing Industrial Cyber-Physical Systems: Multiagent Systems Entering Industry 4.0. *IEEE Ind. Electron. Mag.* **2020**, *14*, 18–32. [CrossRef]
11. Zheng, X.; Psarommatis, F.; Petrali, P.; Turrin, C.; Lu, J.; Kiritsis, D. A quality-oriented digital twin modelling method for manufacturing processes based on a multi-agent architecture. *Procedia Manuf.* **2020**, *51*, 309–315. [CrossRef]
12. Nie, Q.; Tang, D.; Zhu, H.; Sun, H. A multi-agent and internet of things framework of digital twin for optimized manufacturing control. *Int. J. Comput. Integr. Manuf.* **2021**, 1–22. [CrossRef]
13. Lorente, Q.; Villeneuve, E.; Merlo, C.; Boy, G.A.; Thermy, F. Development of a digital twin for collaborative decision-making, based on a multi-agent system: Application to prescriptive maintenance. *INCOSE Int. Symp.* **2022**, *32*, 109–117. [CrossRef]
14. Arabia, C.; Lützenberger, M.; Albayrak, S. Towards adaptive multi-robot systems: Self-organization and self-adaptation. *Knowl. Eng. Rev.* **2018**, *33*, e16.
15. Gascuen, J.; Navarro, E.; Fernandez-Caballero, A. Model-driven engineering techniques for the development of multi-agent systems. *Eng. Appl. Artif. Intell.* **2012**, *25*, 159–173. [CrossRef]
16. Bures, T.; Gerostathopoulos, I.; Hnetynka, P.; Keznikl, J.; Kit, M.; Plasil, F. DEECO: An ensemble-based component system. In Proceedings of the 16th International ACM Sigsoft Symposium on Component-Based Software Engineering, Vancouver, BC, Canada, 17–21 June 2013; ACM: New York, NY, USA, 2013; pp. 81–90.

17. GarcıSanchez, F.; Valencia-Garcıa, R.; Martınez-Bejar, R.; Fernandez-Breis, J.T. An ontology, intelligent agent-based framework for the provision of semantic web services. *Expert Syst. Appl.* **2009**, *36*, 3167–3187. [CrossRef]
18. Bao, Q.; Zhao, G.; Yu, Y.; Dai, S.; Wang, W. Ontology-based modeling of part digital twin oriented to assembly. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2020**, *236*, 16–28. [CrossRef]
19. Van der Valk, H.; Haße, H.; Möller, F. Archetypes of Digital Twins. *Bus. Inf. Syst. Eng.* **2021**. [CrossRef]
20. Eramo, R.; Bordeleau, F.; Combemale, B. Conceptualizing Digital Twins. *IEEE Softw.* **2021**, *39*, 39–46. [CrossRef]
21. Dalpiaz, F.; Chopra, A.; Giorgini, P. Adaptation in open systems: Giving interaction its rightful place. In *Conceptual Modeling—ER 2010*; Lecture Notes in Computer Science; Springer: Berlin, Germany, 2010; pp. 31–45.
22. Dalpiaz, F.; Giorgini, P.; Mylopoulos, J. Adaptive socio-technical systems: A requirements-based approach. *Requir. Eng.* **2013**, *18*, 1–24. [CrossRef]
23. Chopra, A.; Christie, V.; Singh, M. An Evaluation of Communication Protocol Languages for Engineering Multiagent Systems. *J. Artif. Intell. Res.* **2020**, *69*, 1351–1393. [CrossRef]
24. Chavhan, S. Emergent Intelligence: A Novel Computational Intelligence Technique to Solve Problems. In Proceedings of the 11th International Conference on Agents and Artificial Intelligence (ICAART 2019), Prague, Czech Republic, 19–21 February 2019; pp. 93–102. [CrossRef]
25. Harwell, J.; Lowmanstone, L.; Gini, M. Demystifying Emergent Intelligence and Its Effect on Performance in Large Robot Swarms. In Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, 9–13 May 2020; IFAAMAS: Auckland, New Zealand, 2020; pp. 474–482.
26. Chavhan, S.; Venkataram, P. Emergent Intelligence Based QoS Routing in MANET. *Procedia Comput. Sci.* **2015**, *52*, 659–664. [CrossRef]
27. Amelin, K.; Granichin, O.; Sergeenko, A.; Volkovich, Z. Emergent intelligence via self-organization in group of robotics devices. *Mathematics* **2021**, *9*, 1314. [CrossRef]
28. Granichin, O.; Granichina, V.; Erofeeva, A.; Leonova, A.V.; Senov, A.A. Emergent Intelligence and Distributed Stochastic Optimization. *IOP Conf. Ser. Mater. Sci. Eng.* **2022**, *1215*, 1–7. [CrossRef]
29. Rzevski, G.; Skobelev, P. Emergent Intelligence in Large Scale Multi-Agent Systems. *Int. J. Educ. Inf. Technol.* **2007**, *1*, 64–71.
30. Prigogine, I. *The End of Certainty: Time, Chaos and the new Laws of Nature*; Free Press: New York, NY, USA, 1997; p. 240.
31. Kaufmann, S. *At Home in The Universe: The Search for the Laws of Selforganization and Complexity*; Oxford University Press: New York, NY, USA, 1995; p. 336.
32. Prigogine, I. *Is Future Given? World Scientific*; Chiron Media: Wallingford, UK, 2003; p. 160.
33. Shoham, Y.; Leyton-Brown, K. *Multi-Agent Systems: Alghoritmic, Game Theoretic and Logical Foundations*; Cambridge University Press: Cambridge, UK, 2009. Available online: http://www.masfoundations.org (accessed on 20 March 2022).
34. Easley, D.; Kleinberg, J. *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*; Cambridge University Press: Cambridge, UK, 2010. Available online: http://www.cs.cornell.edu/home/kleinber/networks-book/ (accessed on 15 March 2022).
35. Popper, K. *Conjectures and Refutations: The Growth of Scientific Knowledge*; Routledge and Kegan Paul: London, UK, 1963.
36. Skobelev, P. Towards Autonomous AI Systems for Resource Management: Applications in Industry and Lessons Learned. In *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection. PAAMS 2018*; Demazeau, Y., An, B., Bajo, J., Fernández-Caballero, A., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2018; Volume 10978. [CrossRef]
37. Grachev, S.; Zhilyaev, A.; Laryukhin, V.; Novichkov, D.E.; Galuzin, V.A.; Simonova, E.V. Methods and Tools for Developing Intelligent Systems for Solving Complex Real-Time Adaptive Resource Management Problems. *Autom. Remote Control* **2021**, *82*, 1857–1885. [CrossRef]