

Article

Autonomous Digital Twin of Enterprise: Method and Toolset for Knowledge-Based Multi-Agent Adaptive Management of Tasks and Resources in Real Time

Vladimir Galuzin ^{1,2}, Anastasia Galitskaya ², Sergey Grachev ^{1,2}, Vladimir Larukhin ^{1,2,3}, Dmitry Novichkov ¹ , Petr Skobelev ^{3,*} and Alexey Zhilyaev ²

¹ Information Technology Faculty, Samara State Technical University, Molodogvardeyskaya Str. 244, 443100 Samara, Russia; vladimir.galuzin@gmail.com (V.G.); sergey.grachev@gmail.com (S.G.); larukhin@gmail.com (V.L.); d.novichkov@kg.ru (D.N.)

² Knowledge Genesis Group, Skolkovo, Bolshoy Bulv. 42, 121205 Moscow, Russia; galitskaya@smartsolutions-123.ru (A.G.); zhilyaev.alexey@gmail.com (A.Z.)

³ Samara Federal Center of Russian Academy of Science, Studenchesky Str., 3A, 443001 Samara, Russia

* Correspondence: petr.skobelev@gmail.com; Tel.: +7-902-372-3202

Abstract: Digital twins of complex technical objects are widely applied for various domains, rapidly becoming smart, cognitive and autonomous. However, the problem of digital twins for autonomous management of enterprise resources is still not fully researched. In this paper, an autonomous digital twin of enterprise is introduced to provide knowledge-based multi-agent adaptive allocation, scheduling, optimization, monitoring and control of tasks and resources in real time, synchronized with employees' plans, preferences and competencies via mobile devices. The main requirements for adaptive resource management are analyzed. The authors propose formalized ontological and multi-agent models for developing the autonomous digital twin of enterprise. A method and software toolset for designing the autonomous digital twin of enterprise, applicable for both operational management of tasks and resources and what-if simulations, are developed. The validation of developed methods and toolsets for IT service desk has proved increase in efficiency, as well as savings in time and costs of deliveries for various applications. The paper also outlines a plan for future research, as well as a number of new potential business applications.

Keywords: autonomy; digital twin; enterprise; resource management; ontology; multi-agent technology; adaptability; real time

MSC: 68T20; 68T30



Citation: Galuzin, V.; Galitskaya, A.; Grachev, S.; Larukhin, V.; Novichkov, D.; Skobelev, P.; Zhilyaev, A.

Autonomous Digital Twin of Enterprise: Method and Toolset for Knowledge-Based Multi-Agent Adaptive Management of Tasks and Resources in Real Time. *Mathematics* **2022**, *10*, 1662. <https://doi.org/10.3390/math10101662>

Academic Editors: Frank Werner, Alexander A Lazarev and Bertrand M.T. Lin

Received: 21 March 2022

Accepted: 5 May 2022

Published: 12 May 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The concept of “digital twin” [1,2] was introduced not long ago, but it is already expanding very quickly in many directions. Starting with digital shadows and computer models, they have evolved to more complex, adaptive, smart and cognitive digital twins. The future holds intelligent digital twins, which will integrate cyber-physical systems with knowledge bases, machine learning and collective decision-making.

However, in spite of this progress, digital twins have been mainly applied for complex technical objects. The problem of designing digital twins for autonomous management of enterprise resources is still not fully researched.

In this paper, we will introduce digital twins for autonomous enterprise resource management, applying mobile devices for synchronization of orders, tasks and resources of an enterprise with its computer knowledge-based multi-agent model. The proposed autonomous digital twin of enterprise is aimed at implementation of fully autonomous Deming cycle of adaptive allocation, scheduling, optimization, monitoring and control of

tasks and resources in real time. This solution must make this routine work with minimum involvement of humans or, in the future, without them at all.

The autonomous digital twin of enterprise is designed as an intelligent cyber-physical decision-making system which provides convergence of cyber-physical and AI technologies, including ontologies and multi-agent technology. The knowledge-based resource management means that the focus is given on semantic specification of tasks and use of decision-making rules for their matching with required resources. Application of ontologies and knowledge-based reasoning makes it possible to formalize the enterprise domain and create its ontological models, which specify classes of orders, processes and tasks, resources, products and tools, competencies of employees, etc. Application of multi-agent technology makes it possible to provide initial multi-objective planning and adaptive re-planning of tasks to resources by processing events in real time using mobile devices. Multi-agent resource management also means that the system takes into consideration the balance of interests, preferences and constraints of all parties involved in decision making, including not only humans, managers and employees, but also orders, machines, products, equipment, etc.

As a result, an autonomous digital twin of enterprise could be considered as the next step of smart cyber-physical enterprise resource planning (Smart ERP) or, more specifically, advanced planning and scheduling (Smart APS) systems.

The paper formalizes ontology-based multi-agent models and methods for scheduling and optimization of resources, as well as proposing a method and toolset for creating autonomous digital twins of enterprise. The developed models, methods and toolset are applied to different domains, including aircraft and electric cars manufacturing, gas and oil drilling, IT help desk service, etc.

The paper is organized as follows. The second section discusses modern trends in resource management. The third section introduces the concept of the autonomous digital twin of enterprise. The fourth section presents more formally developed ontological and multi-agent models and methods for resource management in the autonomous digital twins of enterprises, which can be customized for a specific enterprise. The fifth section is focused on the method for developing autonomous digital twins of enterprises. The sixth section presents functionality and architecture of the software toolset for developing autonomous digital twins of enterprises. It fully automates business processes of resource allocation, scheduling, optimization, monitoring and control of tasks and resources in real time. The seventh section contains applications of developed method and toolbox for implementing autonomous digital twins of enterprise for different domains, including manufacturing of airplanes, electric cars, gas-oil drilling, etc. The eighth section demonstrates an example of application for IT help desk service and shows the effect and value for business. The ninth section gives an outlook on future research and potential applications.

Application of the developed models, methods and tools is intended to solve complex problems of resource management in real time, reduce peaks, idle-runs and lack of resources, increase business efficiency and decrease man-efforts, time and costs for development and maintenance of autonomous digital twins of enterprise.

2. The Modern View on the Resource Management Problem

Modern enterprise resource management requires high adaptability of scheduling and optimization because of high uncertainty and turbulence on global and local markets, when a number of unpredictable events take place very often and constantly ruin the previously agreed plans. Some examples of such unforeseen events include a new order, broken equipment, an unavailable worker, delay in supply, etc.

The discussed complexity and high turbulent dynamics lead to the fact that traditional, centralized, hierarchically organized, sequential methods and algorithms of combinatorial search or heuristics cannot effectively solve the problem of adaptive resource management with acceptable quality and within the available time.

According to [3], many existing software solutions for resource management (for example, IBM i-Log, SAP, i2, j-Log, Quintiq, Maximal Software, FICO, etc.) are still primarily based on traditional linear, dynamic or constraint programming methods with high computational complexity and a number of restrictions.

To reduce the complexity and provide more efficient search of options in decision making space, a number of new methods are based on heuristics and meta-heuristics, providing near-to-optimal (not fully optimal) solutions of the problem:

- Greedy local search algorithms based on rules of the problem domain;
- Neural networks and fuzzy logic;
- Bio-inspired methods: genetic algorithms, ant colony and particle swarm optimization, etc.;
- Tabu search;
- Simulated annealing;
- Stochastic methods (such as the Monte Carlo method);
- Metaheuristics: combination of heuristic algorithms of optimization, etc.

Unfortunately, these methods and tools also become not fully applicable for modern enterprises. The developed methods and tools work very well in centralized and hierarchic environment, assuming that all orders and resources are given in advance, have the same objectives and do not change during execution of plans. They ignore individual preferences and constraints and do not support real-time networking, including communication, coordination and negotiations for conflict resolving between all participants. However, the progress of cyber-physical systems and mobile devices already made all assets and participants visible in real time and able to participate in decision making.

As a result, in case of unpredictable events, modern enterprises usually involve additional staff, delay their reaction, freeze a bigger number of products in storages, etc. Such extensive and slow reaction further requires the enterprises to cover costs and increase prices for clients. As a result, enterprises end up losing orders and wasting resources, which leads to a decrease of quality, efficiency of services and competitive advantages.

The objective of this paper is to develop new models, methods and toolsets of tasks and resource management for modern enterprises with the main focus on adaptive resource allocation, scheduling, optimization, monitoring and control in real time, when all orders and resources are not given in advance and can change at any time.

3. The Concept of the Autonomous Digital Twin of Enterprise

The concept of digital twin was introduced about 20 years ago in the context of cyber-physical systems and Industry 4.0 [4,5].

At the moment, it is still not fully defined and formalized (the new standard ISO 23247 “Digital Twin framework for manufacturing” is on the way), but in practice, one can define [2] the following three main properties of digital twins:

- (1) Being a virtual representative of the physical object, which can be applied for planning and simulations;
- (2) Providing ongoing self-synchronization between the model and real object;
- (3) Supporting the autonomy of a virtual object compared to the real object.

This vision of digital twin reflects the fast-going convergence of modern information technologies to create holistic virtual model of object, which can operate autonomously, in parallel with the real object.

The number of applications of digital twins is also growing promptly and Gartner includes digital twins in the most perspective technologies [6]. The key software providers, including Siemens, IBM, Oracle, SAP, Autodesk, ANSYS and many others, are currently developing IoT platforms and solutions for creating digital twins of various objects.

The market of digital twin solutions already has an increase of 20% annually [7] and, as expected, it will continue to grow from 3.1 billion USD in 2020 to 48.2 billion USD in 2025 (an approximate 15-fold increase). The number of research papers (according to Web of Science) has increased 10 times in the last 5 years [7].

At the current stage of research and development, digital twins are mainly associated with models of technical objects [1,2,4,5]. However, recently, digital twins have also been applied for enterprise modeling [8–14]. For example, one paper [14] focused on designing digital twins of new post offices in France based on the cooperation of humans and robots. The developed models and methods of digital twins for enterprise modeling include machine learning, simulations, surrogate models, etc.

As a result, digital twins are now considered as a new paradigm of digitalization and automation of enterprises which integrate virtual models of objects (including enterprises) with partially or fully autonomous decision making.

In this paper, we introduce the autonomous digital twin of enterprise as a hybrid knowledge-based multi-agent cyber-physical system which can contain a cyber-physical subsystem, including sensors, computers, communication units and executors, and an intelligent decision-making subsystem, which contains a knowledge base and a multi-agent decision making system, synchronized with enterprise equipment via sensors and with employees via mobile devices.

Autonomous systems are considered goal-driven knowledge-based systems which are able to analyze problems, use knowledge base and plan their activities to solve problems and control results. In the domain of resource management, autonomous systems can provide planning, optimization and control of trucks, factories or supply chains, all without the involvement of people. In the future, autonomous systems will provide unmanned management not only for humans as employees, but also for driverless trucks, fully robotic factories or supply chains, fully working without humans. Autonomous systems integrate many modern information technologies such as cyber-physical systems [15,16], classical planning and optimization [17], multi-agent technologies [18–20], model-driven simulations [21–24], knowledge-based decision making and reasoning [25], etc.

The fast-going convergence of these technologies has resulted in the different concepts and archetypes of autonomous digital twins of things [26,27].

The functionality of the autonomous digital twin of enterprise (autonomous enterprise) for task and resource management will include:

- Ontological specification of enterprise structure, products, business processes and resources in a knowledge base;
- Loading of enterprise model specifications from knowledge base to take into account its characteristic aspects;
- Reaction to events, decomposition of processes to the level of tasks, allocation of tasks to resources, planning and optimization of resources;
- Communication of plans and results with employees;
- Approve and coordinate plans for employees;
- Monitoring and control of plan execution;
- In case of growing gap between plan and reality, adaptive re-scheduling is triggered automatically;
- Evaluation of enterprise productivity and efficiency.

The main steps in synchronization procedure between real and autonomous enterprises are illustrated in Figure 1, where real enterprise is shown on the left side and its autonomous digital twin, which is mirroring its current state, is on the right side.

Processes of synchronization of real and virtual enterprises presuppose the following kinds of communication:

- The flow of new unpredictable events is coming from the real enterprise to the autonomous digital twin of enterprise and each event triggers an adaptation of the plans;
- New allocation, scheduling and optimization of resources take place and new dynamic schedules become available for employees, i.e., managers and workers;
- Managers and workers approve schedules or change them according to their own preferences and constraints;
- The resulting collectively formed schedules are sent to all affected employees as instructions;

- The execution of each task is confirmed by employees via their mobile devices or using factory sensors;
- In case the task is not confirmed in time, the autonomous digital twin of enterprise checks the availability of the employees and starts adaptive re-scheduling of resources or escalates the issue onto the next level for managers.

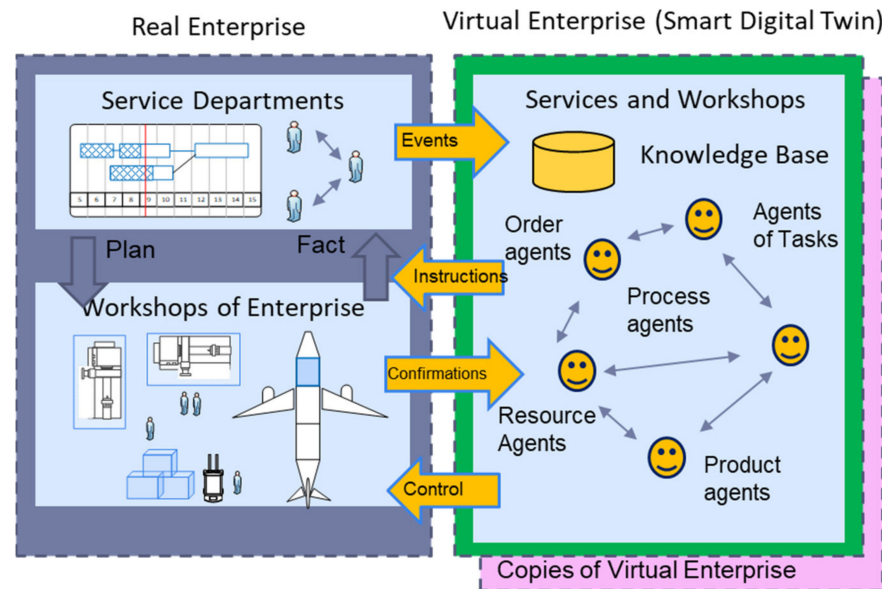


Figure 1. Synchronization between real and virtual enterprises.

Let us define $S_{twin} = \{s_i\}$, where $s_i = (model_i, plan_i, kpi_i)$, where S_{twin} is the state of a digital twin of the actual enterprise, $model_i$ is an ontological model of the enterprise, $plan_i$ is the schedule of orders and resources, kpi_i is a key performance indicators (for example, service level, profit, time of delivery, etc.) and $i = 1, \dots, n$ is the number of states.

Whenever a disruptive event ($Event^{(k)}$) occurs in the actual enterprise, the schedule of the virtual enterprise must change as quickly as possible to a new state in order to achieve adaptation:

$$S_{twin}^{(k+1)} = F(S_{twin}^{(k)}, Event^{(k)})$$

This means that a new $k + 1$ state $S_{twin}^{(k+1)}$ of a virtual enterprise is formed by processing of the new coming event $Event^{(k)}$ by the functionality of the virtual enterprise, which has at the moment the current state $S_{twin}^{(k)}$; F is the functionality of adapting the enterprise schedule in case of event $Event$.

Let us define D as a function describing the difference between the actual and virtual enterprise schedules. It is essential that the state of the actual enterprise S_{real} and the state of its digital twin (virtual enterprise) S_{twin} are always as close as possible.

Then the key objective of the autonomous digital twin of enterprise is to minimize difference in KPIs between real and virtual enterprises at every moment of time k :

$$D(S_{real}^{(k)}, S_{twin}^{(k)}) \rightarrow 0$$

where D is the difference in KPI between real and virtual enterprises, $S_{twin}^{(k)}$ is the state of the virtual enterprise and $S_{real}^{(k)}$ is the state of the real enterprise.

The same description can be granulated to the states of each department, employee, process, task, machine, product, equipment or other resource of the enterprise: let us define $s(O)$ as a state of the object O in the business process of the enterprise.

We assume that all objects in the developed approach will have their own dynamic personal schedules and KPIs. These schedules are not pre-defined and can change depending on events, at any time. For example, the task can be unsatisfied with its KPI, leave the plan of a certain worker or equipment and find a more suitable one for its execution.

The autonomous digital twin of enterprise must be also applicable for real-time simulations of enterprise for modeling such events as modernization of equipment, changing the number of workers, reorganization of daily shifts, etc.

4. Formal Model of Domain Knowledge and Multi-Agent Decision Making in the Autonomous Digital Twin of Enterprise

4.1. Basic and Extended Domain Ontology of Resource Management and Ontological Model of Enterprise

Formalization of collective decision making in the autonomous digital twin of enterprise is based on the application of ontologies and multi-agent technology.

Ontology is defined as conceptualization and formal specification of the domain knowledge [28–30]. Usually, ontology consists of the most generic and abstract classes of concepts and relations which form semantic networks. Instances of concepts and relations form a knowledge base, which can additionally contain rules of reasoning.

One of the first applications of ontologies is Semantic Web, used for the annotation of Internet pages, but it is also applied for various smart services, data management, etc. [31,32]. Recently, ontologies have been applied for specifying manufacturing capacities [33] and digital twins in manufacturing [34], but they were not granulated to the task level of operations.

The idea of using ontologies in this paper is to automate the development of the autonomous digital twin of enterprise by creating and customizing ontological models of enterprises at the level of tasks and resources, which can be loaded into a unified multi-agent system for autonomous task and resource management.

This process requires basic ontology of resource management, which was not available in the literature. As a result of interviews with experts, a review of a number of publications and a systematic analysis of a number of software solutions for resource management, the following basic classes of concepts were extracted: “order”, “business process”, “product” or “service”, “resource”, “tool”, “part” or “material” and some others. Besides such concepts, it will also require basic classes of relations, such as that an order “requires” a business process, a business process “consists” of “tasks”, a task “requires” an employee, an employee “has” competencies, etc. The idea was to select such concepts and relations which can help to specify wide spectrum of situations with resources in real enterprises.

Using these concepts and relations, it becomes possible to create ontological specification of different enterprises in one language (“dictionary”), which is “understandable” for the unified multi-agent system for adaptive resource management.

Let us define ontology as the following set:

$$O = (C, R, F),$$

where C is a subset of object classes or concepts, R is a subset of properties and relations and F is a subset of procedures to operate with concepts and relations, including:

1. Concepts = $\phi_1(c)$ —get a set of all concepts Concepts $\subseteq C$, which are produced from $c \in C$;
2. Relations = $\phi_2(c)$ —get a set of all relations Relations $\subseteq R$, which are produced from $r \in R$;
3. Instances = $\phi_3(c)$ —get a set of all Instances of Class $c \in C$ (including instances);
4. AreRelated = $\phi_4(c_1, c_2)$ —check if concept $c_1 \in C$ is produced from $c_2 \in C$;
5. AreRelated = $\phi_5(r_1, r_2)$ —check if relation $r_1 \in R$ is produced from $r_2 \in R$;
6. IsPart = $\phi_6(i, set)$ —find out if instance i belongs to the given set, by comparing attributes and relations of instance with the given set;
7. Tasks = $\phi_7(p)$ —find all tasks, which produce the given product $p \in ProducedProduct$;

8. Resources = $\phi_8(t)$ —find all resources which are required for the given task $t \in \text{Task}$;
9. Products = $\phi_9(t)$ —find all products for the given task $t \in \text{Task}$.

The developed interpretation Φ provides possibility for more complex requests, for example, Resources = $\phi_8(\phi_7(p))$, to find a set of resources which are required for manufacturing product p .

These components are implemented as Java services and are available for decision making and reasoning of agents.

Let us define an ontology for adaptive resource management as basic ontology, Ob , which contains objects (see Table 1), and its extension, domain ontology, Od , which will contain concepts and relations specific to enterprises, operating in the following domain:

$$Od \supseteq Ob.$$

Table 1. Basic ontology concepts for managing resources.

Order	Specification of the required product, quantity of these products and the time interval of order execution
Product	Specification of products which can be consumed or produced
Task	Specification of input and output objects, next and previous tasks, composition of tasks and required resources for the action
Resource	Specification of human, physical or financial resources required

Ob , the basic ontology, will be supported by unified multi-agent system for adaptive resource management, which will have hardcoded basic classes of agents, providing access to ontology through an interpretation function F .

Extensions of the concepts and relations of domain ontology, Od , are inherited from concepts and relations of the basic ontology, Ob , to establish a link between the two parts of ontology and, thus, enable agents to manage enterprise specifics. If some of the concepts and relations in Od , required for solving domain problems, are not inherited from Ob , it will require introduction of new specific agents in unified multi-agent system for adaptive resource management.

The main idea of the developed ontological models of enterprises can be described in a following way. The designed basic classes of agents (will be described below) are pre-programmed and hardcoded for processing concepts and relations of basic ontology only (order, object, task, resource, etc.). These concepts do not represent any specific domain knowledge, which is always required for enterprises' planning and scheduling. The specifics are given in domain ontology, which is designed as an extension of basic ontology. In the domain ontology, one can specify concrete types of orders, structure of manufactured products, equipment or competencies of workers which are required for scheduling specific objects, for example, to assemble the airplane. Agents read this domain knowledge from the knowledge base, when new order arrives, and apply this knowledge for reasoning and decision-making. For example, the agent of each task can find all types of resources using the "require" relation: equipment, workers, etc. Then, the agent of the task, using the types of required resources, can find concrete equipment and concrete workers, which are in full or partial match with these requirements. As a result, new types of requirements can be added to the knowledge base "on the fly" during computations and do not require system stop and re-programming of the system.

Concepts of the basic ontology described in Table 1 can be formally specified as:

$$Ob = \{\text{Order}, \text{Product}, \text{Task}, \text{Resource}\}.$$

Each order creates a product connected to an appropriate task:

$$\forall x \exists y (\text{Order}(x) \rightarrow \text{Product}(y) \wedge \text{create}(x, y)).$$

The products are separated into produced products and utilized ones. The relation between a task and these two types is given by the following formulas:

$$\forall_x \exists_y (ProducedProduct(x) \rightarrow Product(x) \wedge Task(y) \wedge produce(y, x)),$$

$$\forall_x \exists_y (ConsumedProduct(x) \rightarrow Product(x) \wedge Task(y) \wedge consume(y, x)).$$

Each task can be specified as subsets of the atomic tasks or the group tasks.

The main classes of relations between tasks are “to be part of” and one task “is followed” by another task. The discussed relations help agents find the previous and the next task whenever a need occurs to change the sequence of tasks due to occurrence of a disruptive event:

$$\forall_{x,y} (partof(x, y) \rightarrow Task(x) \wedge Task(y)),$$

$$\forall_{x,y} (follow(x, y) \rightarrow Task(x) \wedge Task(y)),$$

$$\forall_x \exists_y (GroupTask(x) \leftrightarrow Task(x) \wedge Task(y) \wedge partof(y, x)),$$

$$\forall_x (AtomicTask(x) \leftrightarrow \neg GroupTask(x)).$$

The basic classes of tasks are given in Table 2 and classes of resources-in Table 3.

Table 2. Types of basic tasks.

Atomic task with fixed task duration	Specification of the task which must be completed within a specified interval of time
Atomic task with fixed work volume	Specification of the task, the duration of which depends on resources and/or product volume
Atomic task (hammock)	Specification of the task, which must be accomplished in a correct sequence
Composite task	Specification of the task, the duration of which equals the sum duration of sub-tasks

Table 3. Types of basic resources.

Consumable resource	Specification of the resource, which is consumed during the task fulfilment
Reusable resource	Specification of the resource, which is available for the next task immediately after completion of its use

The key class of relations is “require”, which describes the type of resources required for fulfillment of a task; for example, it could be a human with competencies and experience, some equipment or tools and materials.

One task can require many different types of resources of different kinds.

$$\forall_{(x,y)} (require(x, y) \rightarrow Task(x) \wedge (ResourceRequirement(y) \vee Resource(y)))$$

Products may need to be “stored”:

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

A set of basic relations can be formally described as the following:

$$R_b = \{create, consume, produce, partof, follow, require, stored\}$$

Note that it is possible to introduce new concepts and relations in the domain ontology Od , which are linked to the basic ontology, Ob , whenever required, and it will not change the agents' logic and behavior.

This essential feature is enabled through appropriate F interpretations of concepts and relations. For example, in the manufacturing domain, new types of products could be introduced, for example, “components”, “assembly elements” and “final products”. Tasks could be specified as a “process” and “operation”, and resources as “equipment”, “tool” or “employee”, and it will not require changes in agents logic if F provides an opportunity to make the required reasoning:

$$\forall_x (Product(x) \rightarrow Component(x) \vee AssemblyElement(x) \vee FinalProduct(x)),$$

$$\forall_x (Task(x) \rightarrow Process(x) \vee Operation(x)),$$

$$\forall_x (Resource(x) \rightarrow Equipment(x) \vee Tool(x) \vee Employee(x)).$$

The enterprise ontological model, M , is built from the basic ontology, Ob , and domain ontology, Od , as follows:

$$M = \{O_d(O_b), I\},$$

where I is a subset of instances of the previously entered concepts, such as “equipment units” with inventory numbers.

The enterprise scene, S , which represents an instantaneous state of the enterprise and contains values of attributes of all instances of enterprise concepts and relations at time t , is built in the following way:

$$S = M(t). \quad (1)$$

The enterprise scene is implemented as a database of interconnected instances of concepts and relations that enable agents to easily find a task specification and use the topology of complex schedules to rapidly select an appropriate group of agents for every collective decision making, substantially reducing the time required for considering decision options and performing calculations.

Figures 2 and 3 illustrate the basic and domain ontologies, respectively, for a manufacturing enterprise.

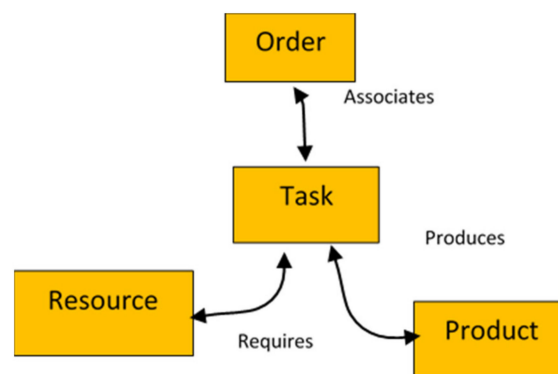


Figure 2. The basic ontology: key concepts and relations.

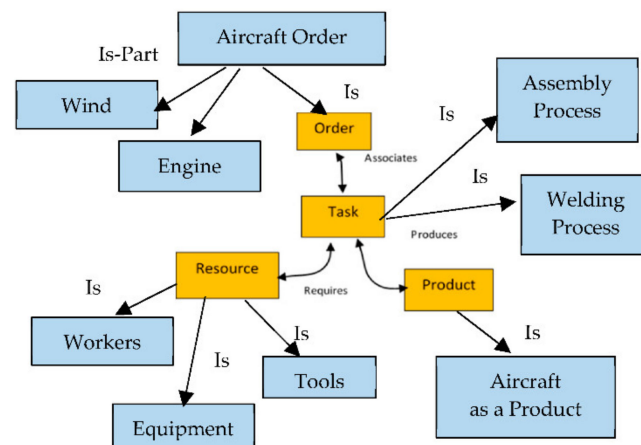


Figure 3. Domain ontology for manufacturing: concepts and relations (in blue) which provide extension to basic ontology (in yellow).

As was already mentioned, the main element of ontologies for any adaptive scheduler is the concept of “task”, which defines what resources are required, which is the previous and next “task” and some other relations.

An example of a concrete task for assembling aircraft MC-21 in the Irkut factory is presented in Figure 4.

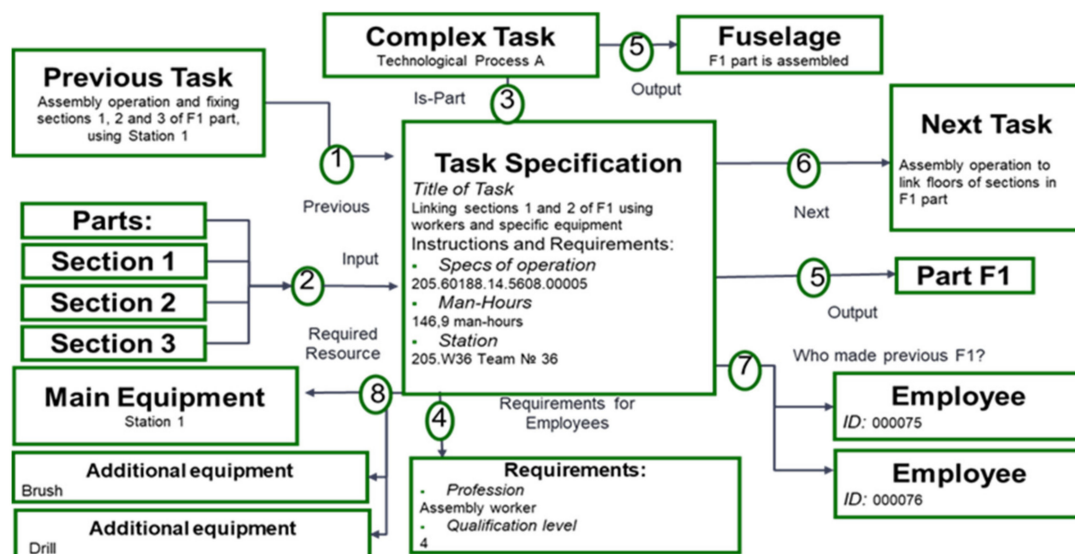


Figure 4. An example of one concrete task for assembling of aircraft MC-21 in the Irkut factory.

The highlighted relations (in circles) play the following roles for task specification:

1. Previous Task—specifies the link to the previous task;
2. Input Objects—specifies the set of objects that can trigger the task in case they all are available;
3. Upper Technology Process—specifies the Technology Process, which must receive the information about successful implementation of planning or execution or to which some issues may be escalated;
4. Required Resources—specifies all human, equipment or other resources required for task planning or execution;
5. Next Task—specifies the link to the next task;
6. Output Objects—specifies the set of resulting objects;

7. Who performed this task in the past?—specifies the person who is an expert in this task implementation and can provide consultation in case of any issues.

The basic agent of the “task” class can read this information and create instances of this agent for this specific task, which will represent interests of this task and work on its behalf. For example, if a delay is identified in the current task, then the agent will find the next task and send a message to it with a warning. This message may trigger re-planning of the next task in case of shortage of time and its re-allocation to some other resources.

The last relation is not implemented in the core part of this basic agent of task and may need additional work for supporting this relation in the unified multi-agent system for adaptive resource management.

The number of such kind of relations for each and every task, hundreds and thousands of which are usually required, helps formalize and utilize domain- and enterprise-specific semantics of tasks for better quality and efficiency of scheduling and optimization.

In this case, the ontology and ontological models specify the directions for agents’ negotiations, significantly reducing the number of options in the decision-making space and the number of computations in the process of conflict resolving.

4.2. Multi-Agent Model and Method of Adaptive Allocation, Scheduling, Optimization, Monitoring and Control of Tasks and Resources

Multi-agent technology is a new paradigm for developing autonomous, distributed and self-organized systems [35]. Multi-agent technology complements digital twin concept by decision making mechanisms and applied for BIM modeling [36], control of quality of manufacturing [37], process optimization [38], collaborative decision making in maintenance [39], etc.

An agent is defined as an autonomic software object which is able to react to events, make decisions and communicate and coordinate these decisions with other agents. However, despite the fact that multi-agent technology is a very attractive paradigm for software engineering, up until recently, it was mainly well known in the academic community. The main reason for this is that multi-agent systems are hard to develop and are lacking adequate models, methods and tools for collective decision-making, particularly in the domain of complex resource scheduling and optimization problems.

The hypothesis of this paper is a new research and development paradigm, in which the solution to any complex problem can be formed by self-organization of goal-driven autonomous agents, which have conflicting objectives but are able to continuously negotiate and solve conflicts by finding trade-offs.

In our previous research, we have already developed multi-agent models, methods and tools for solving various complex problems in industry, including resource management, text understanding, clustering, etc. [40,41]. It was experimentally proved that the developed multi-agent methods for collective decision making provide benefits for adaptive scheduling of resources for transportation, factories, supply chains and logistics.

In this paper, we will make the next step and develop new multi-agent models and methods to cover a full Deming cycle by combining resource allocation, scheduling and optimization with monitoring and control of the states of tasks and resources in real time. The problem is complex enough and requires taking into consideration a lot of domain-specific semantics, which will be represented by ontology-based models of enterprises.

For solving resource management problems, we are proposing the Order-Technology-Process-Task-Product-Resource-Staff agent model (OTPTPRSA-model), which will extend the Product-Resource-Order-Staff-Agent model (PROSA-model) developed in [42]. In the proposed approach, each type of agent has its individual goal, preferences and constraints and is constantly trying to achieve better results. Instances (clones) of basic agents can also have individually defined settings of goals, for example, as a liner combination of service level, time, cost and risk functions [43].

The process of forming plans of resources in the developed approach is based on the Virtual Market (VM) concept, in which agents can buy and sell time slots in the enterprise

schedule. The origins of VM concept can be found in works on electronic auctions, but the most fundamental results are presented in [44,45]. It was proved that VM methods and algorithms in some cases are equal to linear programming, for example, in solving the assigning problem, and many good properties of these methods were identified: flexible, efficient, well parallelized, easy to understand, stable to specification changes, etc.

In the developed VM approach, the main agents are task agents, which are looking for the required time slots of resources and book these time slots, whereas resource agents are looking for the most suitable orders and tasks to cover their costs. Agents of tasks and resources compete and cooperate not only for free time slots, but also for already booked ones. If a conflict is detected, for example, a resource is already occupied by a task, an agent of the task that brings less value to the system gives way, but may receive compensation from the resource or from the system as a whole. This compensation is calculated during new allocation of this task with the use of individual satisfaction and bonus penalty functions. As a result, the problem solution is self-organized in a step-by-step way by collective, parallel and asynchronous processes of detecting and solving conflicts between orders, products, processes, tasks and resources. Developed protocols of solving conflicts among agents implement methods of negotiations for finding elastic trade-offs using penalties and bonuses in very concrete situations. The solution is found when a “competitive equilibrium” (“consensus”) is reached and no agent can improve its KPIs in the given situation. Due to self-organization by autonomous collective decision-making principle, the whole schedule can flexibly adapt itself in case of unforeseen events.

Let us assume that VM is triggered by disruptive events coming from the real enterprise. The purpose of the multi-agent system for adaptive resource management will be to minimize the negative consequences of this disruption and achieve this goal by initiating a wave of changes in the self-organized schedule. This wave triggered by the first affected agent that received information on disruption. Decisions on the change are made through negotiation of affected agents to resolve conflicts and achieve a new consensus.

As was stated above, the enterprise state is defined as a sum of states of all objects and agents participating in the enterprise technology or business processes, including orders, technology or business processes, tasks, products, human resources, equipment and materials. Whenever a disruptive event occurs in the actual enterprise, the schedule of the virtual enterprise must change as quickly as possible to a new state.

For every basic ontological object, s_i , a corresponding digital agent, a_i , exhibits the object's behavior, as shown in Table 4.

An instantiation of the developed agent classes for the example of the factory workshop is presented in Figure 5.

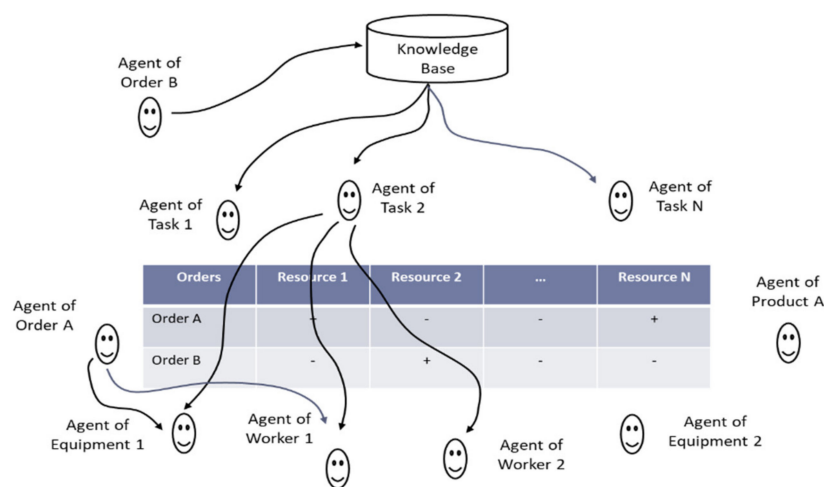


Figure 5. An instantiation of the agent classes for the example of the factory workshop.

Table 4. Objectives, preferences and constraints of the agents.

Agent Type	Objectives, Preferences and Constraints	Attributes
Order agent	To be realized with minimum delay, c , and cost, p , $Y_i = w_1 \left(1 - \frac{c}{c_{kp}}\right) + w_2 \left(1 - \frac{p}{p_{kp}}\right)$	Deadlines, volume, unit costs
Technology/Task agent (atomic, composite)	To be realized using appropriate resource, before deadline, in time $(\tau_i = finish_i - start_i)$ $Y_i = \begin{cases} 1, & \tau_i < \tau_{opt} \\ \frac{\tau_i - \tau_{kp}}{\tau_{opt} - \tau_{kp}}, & otherwise \end{cases}$	Features of resources, products; Deadlines for beginning and ending; connections with other tasks
Resource agent (humans, equipment, etc.)	To be engaged to its max capacity To minimize idle and readjustment times $Y_i = \begin{cases} 0, & u_i < u_{kp} \\ \frac{u_i - u_{kp}}{u_{opt} - u_{kp}}, & otherwise \end{cases}$	Schedule, periods of unavailability, rules for servicing and readjustments, productivity
Product agent	To arrange its own storage To minimize intervals between production and consumption, e $Y_i = 1 - \frac{e_i}{e_{kp}}$	Storage specifications, the time required for delivery, production or consumption
Enterprise (staff) agent	Coordination of agent activities	Planning time, depth of modification chains

Figure 5 shows that new Order B is arriving when Order A has already successfully been scheduled. Order B reads the technological process from the Knowledge Base and created agents of Tasks 1, 2, ..., N. Agents of the tasks start looking for required resources and a new process of agent collective negotiations is triggered. As a result, two conflicts between Order A and Order B are discovered on workers and equipment. The conflicts can be solved by shifts, swaps and drops of tasks with the view on objectives of these and other agents.

Objectives for every software agent are defined using an agent satisfaction function, $Y_i(plan_i)$, which is specified as a linear combination (weighted sum) of M elements belonging to kpi_i and calculated based on the current schedule, $plan_i$, related to the object agent as the following:

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

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

As discussed, the task agents on the VM can purchase time slot from resource agents and resolve conflicts in case several task agents request the same usage time from a resource agent by paying compensation to agents giving up their requests. For this reason, agents can use bonuses awarded for good performance or provide fines for underperforming. Performance is measured by its satisfaction function defined above. For this purpose, each agent has a bonus (fine) function, $B_i(Y_i)$. Virtual money received or expected can be used to compensate to those agents that are losing in negotiations.

VM can have diverse agents with different satisfaction and bonus/fines functions and a facility that allows for an agent instance assigned to every enterprise ontology object instance. Functions of satisfaction and bonuses/fines are introduced to motivate agents to perform as close as possible to their specified KPIs. Resource agents have an additional feature, a cost function, related to the cost of tasks.

Allocation of resources to orders is done in the following way:

1. Following the current state of the abstract world, S_{twin} , instances of order agents, resource agents and product agents are created and receive permission from the enterprise agent to act and take decisions.
2. Agent of an active order, A_k , picks up from the knowledge base the business process for the appropriate product and triggers the required task agents that are connected by nesting or sequencing relations.
3. A high-level task agent checks that relevant products and resources are available and ensures task performance in the specified time.
4. The task solution process starts from generating options by analyzing required resources, comparing task requirements and resource capabilities, resolving resource access timing issues and selecting by the branch and bound method.
5. The computation is based on identification of the set of conflicting orders which substantially reduce the number of solution options:

$$\{a_i \mid i \neq k, plan'_k \cap plan_i \neq \emptyset\},$$

6. The next step is to allocate resources to tasks. Constituent agents allocate resources following the procedure described above. The results are delivered to the relevant group agents, which may accept or reject them and, in the latter case, request a new solution.
7. The group reports to the order agent on the proposed allocation of resources.
8. The order agent negotiates with conflicting orders to resolve conflicts.

The resulting chain of schedule modifications produces losses suffered by agents who agreed to change their requirements to resolve the conflict ΔB_i .

9. This chain of modifications is successfully dimmed if the corresponding order agent can compensate losses of conflicting agents from the gains earned by its bonuses, ΔB_k :

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

10. If this is the case, the schedule is accepted; if not, a new round of negotiations is performed.
11. The order agent then identifies all products linked to it by the “produces” relation and informs their agents when they must be delivered to appropriate stores.
12. The activity ends when a consensus is reached; in other words, 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 ΔB_k or when the time available for negotiations runs out:

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

13. Once a consensus is reached, the VM stops working and is switched to a standby mode, awaiting the next disruptive event.

The fragment of the discussed scheme is illustrated in Figure 6, where it is shown how agents adaptively change the schedule.

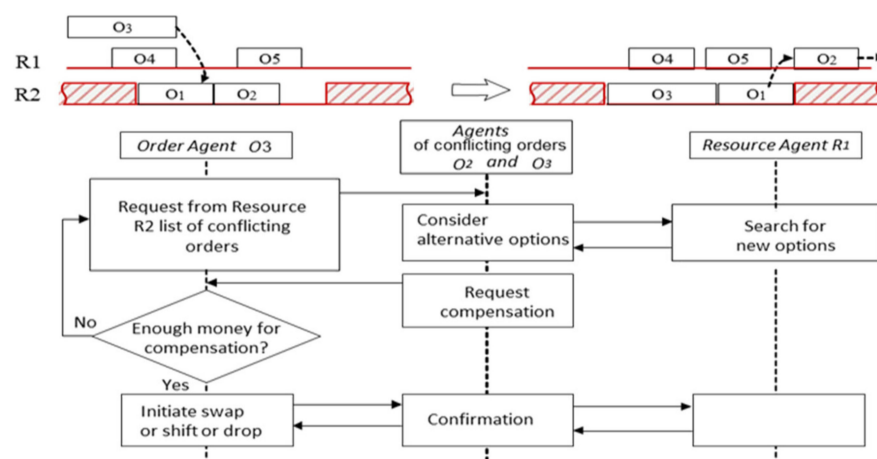


Figure 6. A fragment of agent negotiations for resolving ‘task-resource’ conflicts.

The time required to find the optimal schedule is substantially reduced if the order agents only resolve conflicts in an adaptive manner without full combinatorial search.

If changes in the basic ontology are required, the VM must be re-developed. In contrast, no re-developing is needed if only domain ontology is expanded.

5. The Method of Designing Autonomous Digital Twins of Enterprises for Adaptive Task and Resource Management

The developed method for designing Autonomous Digital Twins of Enterprises for adaptive task and resource management has the following procedure.

1. In the first step, the basic ontology is formalized with the help of experts in ontology editor and basic functionality is implemented in a unified multi-agent system for adaptive resource management covering the Deming cycle.
2. In the second step, the basic ontology is extended by domain-specific concepts and relations for different domains of applications. For example, for the aviation industry, one can introduce aviation ontology, which will specify typical parts of an aircraft, basic assembly processes and tasks, competencies of workers, etc. For the gas and oil drilling industry, the domain ontology will specify typical oil equipment, technological processes, etc.
3. In the third step, the domain ontology can be applied for building ontological models of concrete enterprises specifying its departments, business processes, persons and their roles and competencies, etc. Two different enterprises, working in one domain, can have the same domain ontology or can expand and modify their copies of common domain ontology to make it more suitable for their business aspects.
4. Ontological models of enterprise can be loaded into the unified multi-agent system for resource management. A set of such models can form a collection in the knowledge base for the domain of enterprises and play a role of standards in future.
5. To launch the Autonomous Digital Twins of Enterprises, users only need to load the selected ontological model of enterprise and to specify the initial scene or state of the enterprise, including the values of object attributes.

As a result, the proposed method provides the opportunity to formalize domain-specific knowledge of enterprises, which is usually “out of consideration” in traditional methods and tools of planning and optimization. For example, ontological specifications of the domain can help take into consideration what kind of resources is required for this particular task (operation), what input objects can trigger this task, what is the previous and the next task, etc.

The structure of ontological specifications of the method is given in Figure 7.

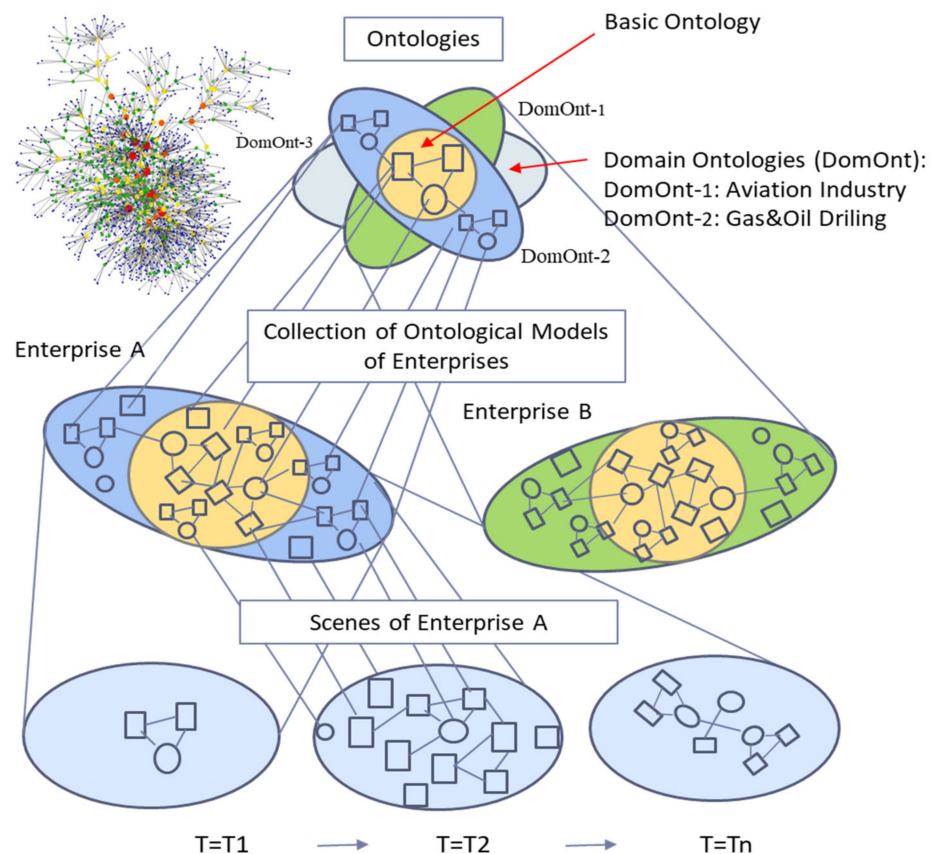


Figure 7. Structure of the ontological specifications of the enterprises.

In this case, this domain-specific knowledge and logic can be separated from the source code of the system, which will become more generic and unified for various applications.

There is a number of ontologies developed for many different domains, including manufacturing, transport or agriculture.

However, applications for task-centric resource management and customization of multi-agent systems are not yet known and researched.

When the ontological enterprise model is loaded, agents make copies (clones) of their basic classes and use the formalized knowledge from ontology to specify behavior of agents of each task; for example, under what conditions can each task be launched, what are the previous and next tasks, what kind of resources this task requires, etc.

The design of a typical agent, is presented in Figure 8, including the components:

- State machine of agent—the set of states and transition rules which are connected and triggered by scene events or messages from other agents;
- Agent services—a number of software components to get input data, make computations, store data, etc.;
- Communication tools—software components to send/receive messages;
- Access to Knowledge Base—software components for accessing the Knowledge Base, reasoning and navigation through semantic networks.

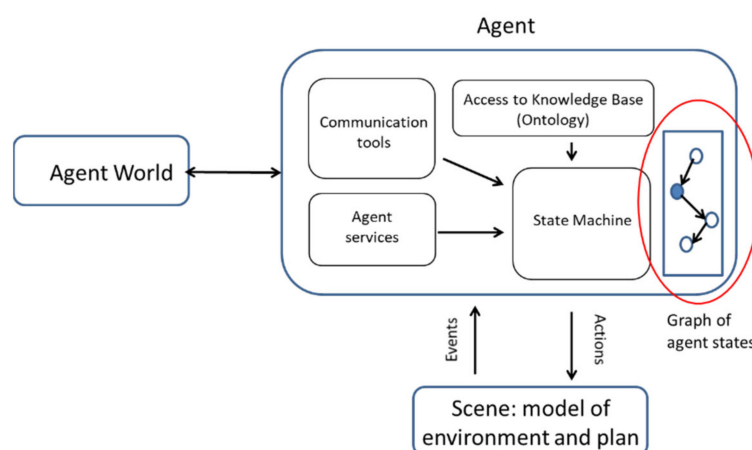


Figure 8. Typical agent structure.

Agent World is a software component for running and dispatching instances of agent classes, for example, the schedule of middle-size workshop with 300 equipment units and 150 workers for the 3–6-month horizon may require more than 5 thousand interconnected instances of Task Agents, and all these instances of agents will be individually customized by the ontological model of enterprise.

The other core part of the autonomous digital twin of enterprise is the ontology-driven knowledge base, which will provide knowledge graph methods and tools to collect, digitalize, formalize and systemize domain-specific knowledge of enterprise, including detailed specifications of orders, business or technology processes, tasks and required resources, humans, machines and equipment, competencies of employees, etc. The granularity of formal specifications of business or technology processes will provide possibility to specify each task in enterprise operation and automatically find the required resources more semantically, individually, adaptively and dynamically.

The discussed concept of the autonomous digital twin of enterprise requires modification of the previously developed methods and tools and integration of ontologies to support the full cycle of resource management, with detailed granulation of specifications to each and every task in business or technology processes.

6. Architecture of Knowledge-Based Multi-Agent Toolset for Designing the Autonomous Digital Twin of Enterprise

The described method was implemented in a Knowledge-Based Multi-Agent Toolset for developing the Autonomous Digital Twins of Enterprise.

The architecture of the Knowledge-Based Multi-Agent Toolset includes the following main components (Figure 9):

- Basic Ontology of Resource Management—contains basic classes of concepts and relations for resource management;
- Domain Ontology of Enterprise—contains domain-specific classes of concepts and relations for the concrete enterprise;
- Ontological Models of Enterprises 1, 2, ..., N—fully specified enterprise, including types of orders, technology or business processes, employees and other resources;
- Scenes of Enterprise, synchronized with the state of real enterprise 1, 2, ..., N—ontological models of enterprises with concretization of attributes of properties and relations in the initial moment of time;
- Ontology, Knowledge Base and Scene Editors—software components for modifying ontologies, knowledge bases and scenes;
- Events—event queue for registering input events and sending them for processing in multi-agent system;

- Ontology-Based Multi-Agent System for Adaptive Resource Management—the unified software component for loading ontological models of enterprises;
- User Interface—web interface and mobile application for communication with users and coordinating decisions.

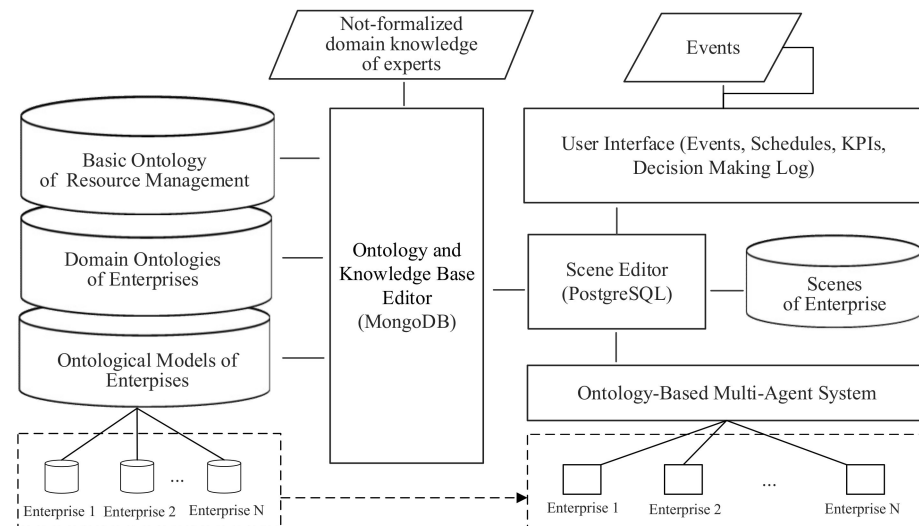


Figure 9. Architecture of a Toolset for Developing Autonomous Digital Twins of Enterprises.

The core parts of Autonomous Digital Twins of Enterprises are the Knowledge Base and Ontology-Based Multi-Agent System for Adaptive Resource Management.

The loading of the Ontological Model of Enterprise in the Ontology-Based Multi-Agent System triggers the creation of a new virtual world, which represents and mirrors tasks and resources of the real enterprise. In this sense, the developed Multi-Agent System is unified for a wide range of enterprises and it significantly reduces the time and cost of the developments.

At any moment of time, the Ontology-Based Multi-Agent System can be copied and the separate scene can be used for real-time simulations and what-if games; for example, entering new potential order, changing technological processes or shifts of workers, modernization of equipment, etc.

The Autonomous Digital Twins of Enterprises can be delivered to client as a stand-alone software solution or it can be integrated as a service with any other digital platforms or the cyber-physical system.

The developed tools can speed up development and save human effort, time and costs of developments and maintenance of the solution.

7. Evaluation of the Efficiency of the Method and Toolset for the Autonomous Digital Twins of Enterprises Development Process

To evaluate efficiency of the discussed method and toolset for developing Autonomous Digital Twins of Enterprises, the following resource management applications were considered:

- Aircraft assembly process for MC-21 for the Irkut Corporation (Irkutsk, Russia);
- Assembly of electric cars using robotic line for the TPA Company (Saint Petersburg, Russia);
- Oil drilling process for the Gazpromneft-Yamal enterprise (Saint Petersburg, Russia);
- Wheat growing for the Rassvet agrofarm (Rostov region, Russia);
- Group of satellites for Earth remote sensing for the Rocket and Space Corporation “Progress” (Samara, Russia).

The Autonomous Digital Twins of Enterprises were prototyped independently for each of these domains, including collecting and formalizing domain knowledge, creating

an ontological model of enterprise and implementing modifications of basic ontology and a unified multi-agent system.

The idea of experiment was to measure the scope and total amount of development work (man-hours), including ontological and multi-agent features only:

- Changes in basic ontology (if needed);
- Extension of domain ontology;
- Changes in agent class logic;
- Addition of new agent classes.

During the experiment, the number of agents and size of ontologies was measured.

Aircraft assembly process for MC-21 for the Irkut Corporation

Input data: set of orders and list of events, in real time, storage of products.

Knowledge Base: product structure breakdown, technological processes with specification of each task, matching rules between tasks and resources, equipment, materials, number of workers and their competencies.

Criteria: just-in-time order production, minimization of order execution time and usage of resources, balanced load of workers.

Assembly of electric cars using robotic line for the TPA Company

Input data: set of orders and list of events, coming in real time, storage of products, plan and tariff of DHL deliveries of car parts.

Knowledge Base: product structure breakdown, technological processes with specification of each task, matching rules between tasks and resources, equipment, materials, number of robots and their functionality.

Criteria: just-in-time order production, minimization of order execution time and usage of resources, balanced load of robots.

Oil drilling process for the Gaspromneft-Yamal enterprise

Input data: requirements for oil drilling, storage of required materials.

Knowledge Base: best practices and technological processes of oil drilling with specification of each task, matching rules between tasks and resources, equipment, materials, number of workers and their competencies.

Criteria: minimization of oil drilling time and usage of resources.

Wheat growing for the Rassvet agrofarm

Input data: wheat variety, the number of fields and days of wheat seeding, weather forecast and actual data.

Knowledge Base: stages of plant growth with specification of each phase, matching rules between stages, weather conditions and available resources.

Criteria: minimization of time for stages and maximization of harvest forecast.

Group of satellites for Earth remote sensing for the Rocket and Space Corporation "Progress"

Input data: the state of satellites and ground stations, set of orders for remote Earth observation.

Knowledge Base: satellites break-down structure, ballistics, technological processes of data imaging with specification of each task, matching rules between tasks and resources of satellites and their equipment functionality.

Criteria: minimization of reaction time and maximization of image resolution.

The results of discussed developments are shown in Table 5.

Table 5. Results of using the developed method and toolset for developing prototypes of Autonomous Digital Twins of Enterprises in different domains (KB—Knowledge Base and MAS—Multi-Agent System).

Domain of Enterprise	Size of Basic Ontology	Size of Domain Ontology	Size of Ontological Model of Enterprise	Number of Agents	Development Time (Man * Months)	
					KB	MAS
Aircraft Assembly	61	152	925	>350	3	3.5
Electro-Cars Assembly		89	382	>520	1	2
Oil Drilling		85	441	>5000	2	3
Digital Twin of Plant		42	236	>100	1	1
Swarm of Satellites		112	304	>450	1	4

The basic ontology O_b was stabilized with about 60 main concepts and relations. Domain ontologies expand this basic ontology up to 2–3 times. Ontological models of enterprises, which include instances of concepts and relations, may differ very much and include from 236 to 925 instances. These ontological models were loaded into the unified multi-agent system and automatically created instances of basic classes of agents. The required development and customization of the unified multi-agent system for these new applications took about 2–4 months.

Compared with the traditional approach for developing smart solutions, when the process of development takes 9–12 months [35], the application of the developed method and toolset makes it possible to significantly (up to 3–4 times) reduce complexity, cost and time of development process.

More detailed results and examples of applications can be found in [46].

8. Example of the Method and Toolset Application

Let us consider an example of the application of the presented method and toolset for the development of the autonomous digital twin of enterprise for the Service Desk Center of the Russian National Railways.

The Service Desk Center serves about 2500 information systems by 4500 employees, which have competencies in different software solutions.

At the moment, the Service Desk Center of the Russian National Railways has an old-fashioned IT system which is planned to be replaced by a more advanced and intelligent decision-making system for scheduling employees and chat bots for users.

The existing system has a module of auto-allocation of new orders to employees in an empiric manner “First In—First Out” and up to 10 orders per employee (other orders are staying in the queue and can lose their service level). This module has no adaptability, so new orders cannot change allocation of the previously allocated resources. Moreover, this module serves orders without any knowledge about semantics of the detected problem, service level for specific system, time zones, competencies of employees and their planned load and other preferences and constraints.

The autonomous digital twin of enterprise for the Service Desk Center was proposed as a solution of the problem and the pilot project was started in the summer of 2021.

The developed new system reads orders and creates their semantic descriptors, analyzes its content and adaptively schedules and optimizes tasks more individually, taking into consideration all the above parameters, including employee competencies. However, it also combines scheduling and optimization with monitoring and control of task execution. For example, if one task is significantly delayed by an employee and has a risk to break services, the next tasks in the schedule of this employee will automatically become active and start searching for other options. As a result of such negotiations for solving conflicts, these tasks can be reallocated and re-scheduled to other employees, for example, who fulfilled the planned tasks faster than expected, and have the needed spare time and competencies.

The difference between two systems is shown in Figure 10: in the existing system (left), new orders are staying in the queue waiting for allocation. In the new system (right),

new orders are immediately adaptively allocated and scheduled by solving conflicts in the schedules of employees. Red circles here identify the detected conflicts and re-scheduling of the previously scheduled orders.

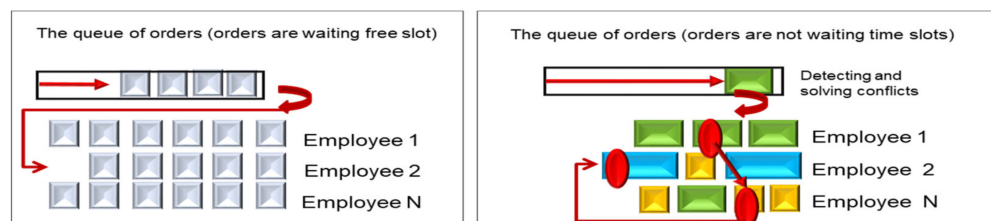


Figure 10. The difference between the existing (left) and developed (right) systems.

The prototype of the autonomous digital twin of the Service Desk Center was developed in 3 months and researched for one group of 34 employees working with 25 software systems in the domain of logistics.

To evaluate the efficiency of the solution, the data records from Service Desk Center for Q1 2021 were collected and analyzed.

The basic ontology was applied and domain ontology and ontological model of the selected logistics group of employees were created, which includes about 100 technological processes (Figure 11) and the matrix of employees' competencies (Figure 12).

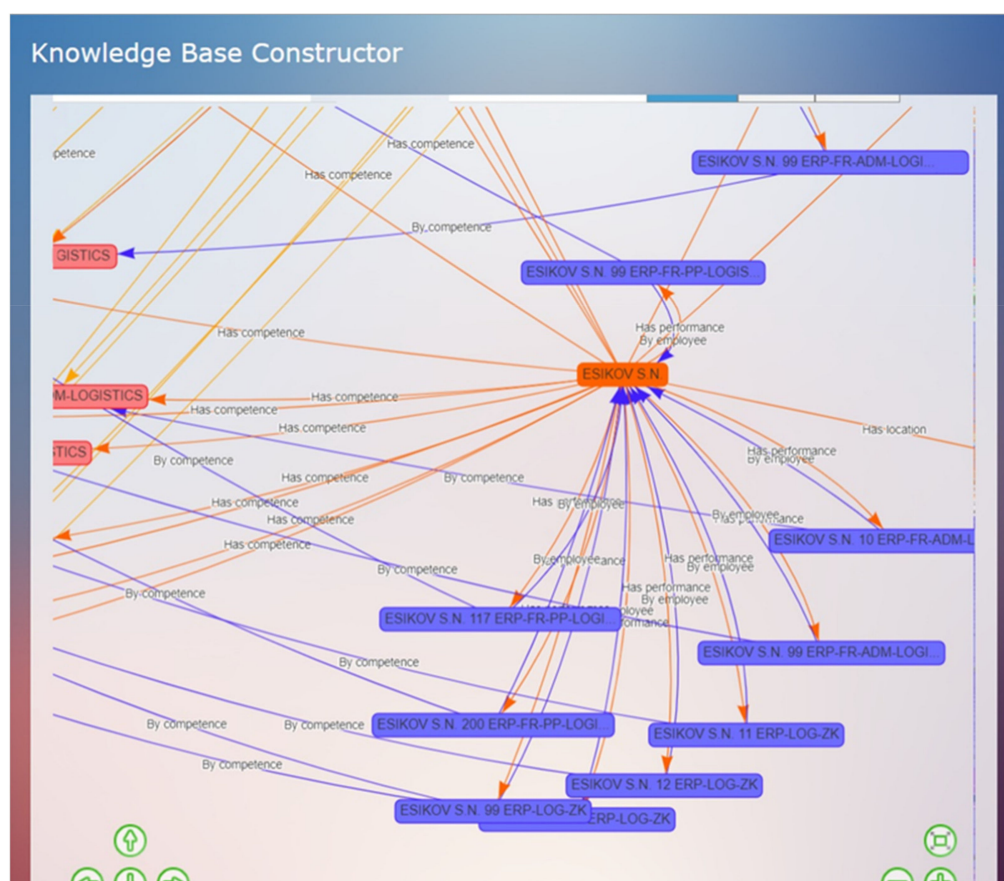


Figure 11. A fragment of domain ontology and ontological model of logistics group of employees (yellow—issues of IT systems, green—technological processes, red—tasks).

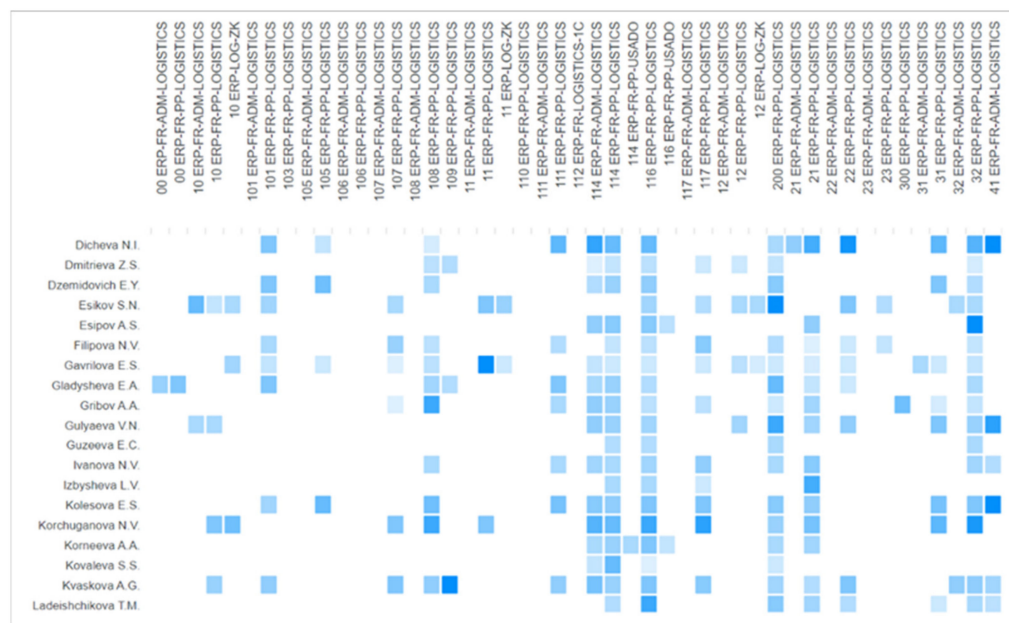


Figure 12. The fragment of competencies matrix (darker blue color—better employee competencies and faster task implementation).

The validation of the developed solution was made with the use of pre-selected real data on 1 September 2021. The test run included 34 employees from the selected team, which processed 304 orders in about 5 h.

The adaptive schedule of employees of the logistics group is presented in Figure 13. One can see here that every employee has tasks with the same color. It means that the system schedules only those tasks that match with the competencies of this employee.

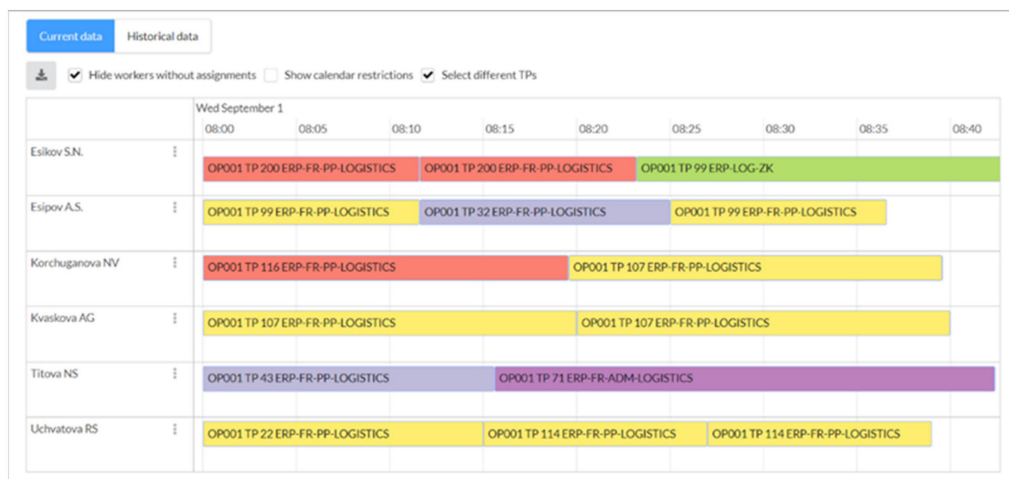


Figure 13. Adaptive schedule of employees of the logistics group.

Figure 14 shows how multi-agent technology works by presenting the satisfaction function and bonus-penalty functions of the system (as a whole), the number of agents involved and the number of changes in the schedule.

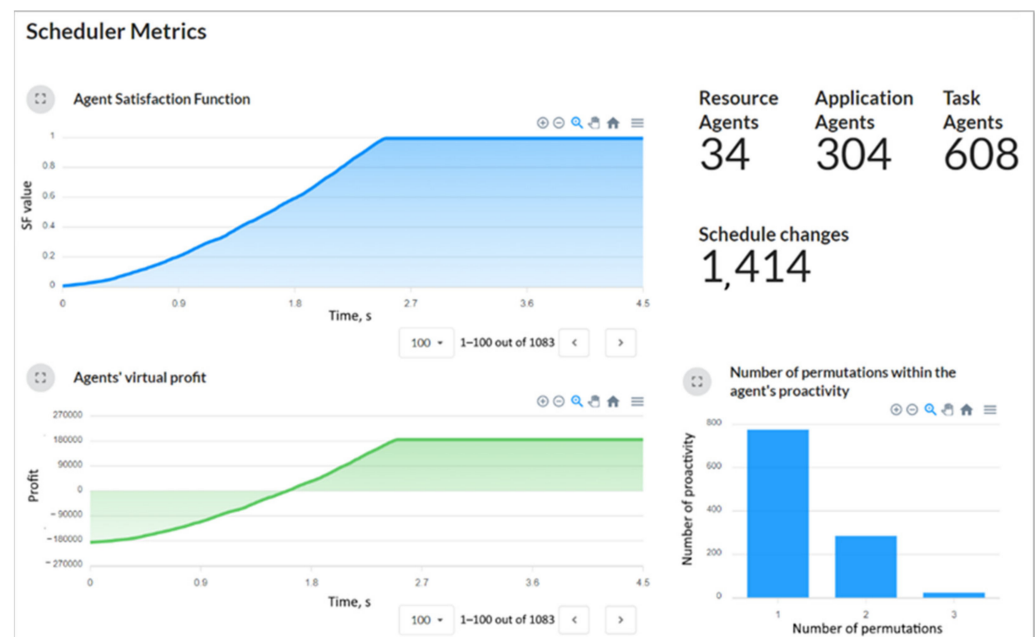


Figure 14. Characteristics of the multi-agent decision-making process.

Here, 304 agents of tasks were allocated to 34 agents of employees. During the negotiation process, the schedule was changed 1429 times and some tasks were allocated at the first attempt, while other tasks changed their positions a few times.

Figure 15 presents results of work of the existing system versus the new one and helps compare these results and evaluate the advantages of the new system.

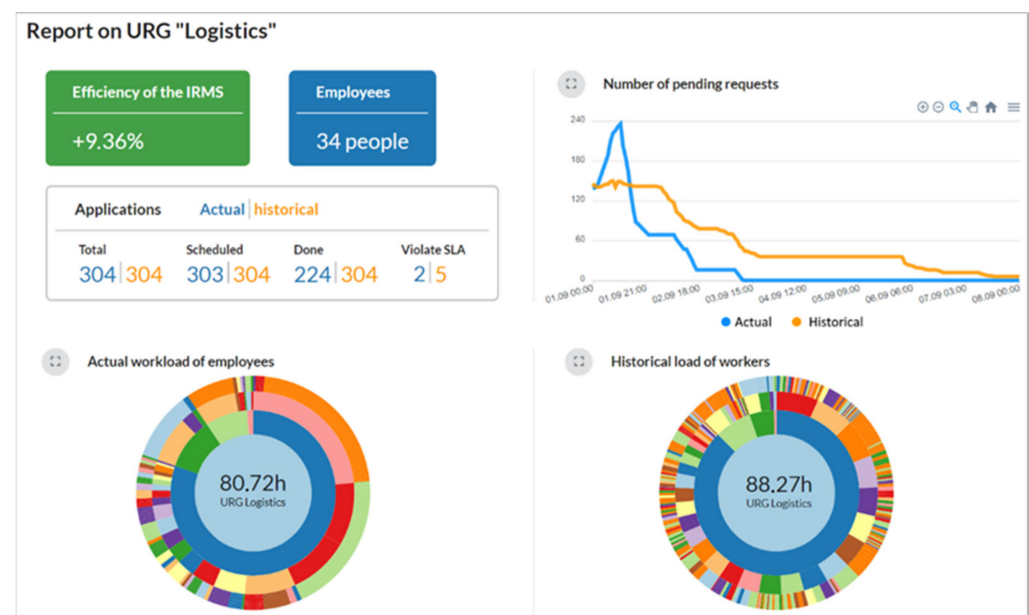


Figure 15. Comparison of results of the existing system and the new one.

The left diagram shows that the developed solution provides an increase of 9.36% compared to the existing solution for resource allocation: the new solution requires 80.72 h of work to process requests and the old solution requires 88.27 h for the same requests. The right diagram shows that the new solution (blue) processes many more orders in the beginning compared to the existing one (yellow). One can see that the new solution has a bigger number of pending requests in the beginning of a day, i.e., the 1st of September

(blue line on diagram), compared with the existing scheduler (yellow line). This means that the scheduler found and scheduled employees with required competencies for solving the problem. Moreover, at the end of the day, the biggest part of all requests was successfully executed and the rest of the scheduled (pending) requests was much lower. This means that the found employees have a bigger productivity and efficiency and are able to process the number of requests much faster than in the existing solution.

The external radiuses of circle diagrams show what tasks are scheduled (in different colors) and their allocation to employees, teams and departments accordingly (internal radiuses). The achieved resource allocation now (on the left) is not so random and fragmented (on the right)—the new solution performs allocation and scheduling of orders more smoothly, using the semantics of the requests and the competencies of employees.

The results of the prototype were analyzed and a number of new opportunities were identified, specifically how to provide a better service for customers, increase the productivity and efficiency of staff, motivate employees to acquire new competencies and reduce risks and penalties.

9. The Discussion and Outlook of Future Developments

The main contribution of this paper is to propose the autonomous digital twin of enterprise based on ontology and multi-agent technology for implementing a full Deming cycle of adaptive resource management, including real-time resource allocations, planning and optimization, monitoring and control of plan execution.

The developed ontology-driven multi-agent approach and software toolset for designing autonomous digital twins of enterprises provides high adaptability for businesses and increases the efficiency of resources under conditions of high uncertainty and turbulent dynamics of demand and supply. The implementation of domain ontology can be done by knowledge engineers for customization of the solution for different enterprises, and this process does not require re-programming of the unified multi-agent system. In case of new types of enterprises, the multi-agent system needs to be modified by new classes of agents and protocols of their communications.

The limitations of the approach are related to egoistic behavior of agents (myopia), which can block the system in local optimums, but this is mitigated by mutual compensations. As a next step, the following modifications can be made: advanced virtual market models, non-deterministic collective decision making with dynamically balanced positive and negative feedback mechanisms, proactivity of both demand and resource agents, etc.

Machine learning/data-driven methods could also be applied for digital twins of enterprises when big data are available and in a situation where orders and resources are not changing during the time period. However, if an enterprise does not have big data or the if situation with orders and resources is continuously changing, these methods cannot be fully applied. However, in the future, these methods could be efficiently combined in a new hybrid method.

Another issue of the developed approach is the scalability for large enterprises and supply chains. The solution could be based on holonic and network-centric architectures for cooperation and competition of autonomous digital twins of enterprises. For example, the autonomous digital twin of one workshop can interact with the autonomous digital twin of another workshops. Autonomous digital twins of workshops can populate autonomous digital twins of factory, etc.

The plans for future research and developments include the following:

- Analysis of non-deterministic decision making in self-organized multi-agent systems and phenomena of emergent intelligence;
- Collection of pre-built ontologies and ontological models of enterprises and basic classes of agents for a wide range of different domains;
- Distributed knowledge base for supporting automatic decision making in specific teams of employees;

- Digital platform and digital ecosystems for p2p competition and cooperation between Autonomous Digital Twins of Enterprises;
- Integration with chat bots and software robots for business processes;
- Integration with Internet of Things (IoT) platforms.

The autonomous digital twin of enterprise can be applied for automatic resource management synchronized with real enterprise or as a real-time simulation tool.

The results could be used for academic research and industry, including potential business applications for project management, agile programming, remote work from home in pandemic times and other cases.

Author Contributions: Conceptualization, P.S.; Formal analysis, V.L.; Investigation, D.N.; Methodology, S.G.; Software, A.G.; Writing—original draft, V.G. and A.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by a grant of the Russian Science Foundation No. 22-41-08003, <https://rscf.ru/project/22-41-08003/>.

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

Data Availability Statement: Not applicable.

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

References

1. Grieves, M.; Vickers, J. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*; Kahlen, F.-J., Flumerfelt, S., Alvesm, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; pp. 85–113.
2. Barricelli, B.; Casiraghi, E.; Fogli, D. A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications. *IEEE Access* **2019**, *7*, 167653–167671. [[CrossRef](#)]
3. Proceedings of the 31st European International Conference on Operational Research, Athens, 11–14 July 2021. Available online: <https://euro2021athens.com/> (accessed on 5 March 2022).
4. Qi, Q.; Tao, F. Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison. *IEEE Access* **2018**, *6*, 3585–3593. [[CrossRef](#)]
5. Tao, F.; Cheng, J.; Qi, Q.; Zhang, M.; Zhang, H.; Sui, F. Digital twin-driven product design, manufacturing and service with big data. *Int. J. Adv. Manuf. Technol.* **2018**, *94*, 3563–3576. [[CrossRef](#)]
6. 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 5 March 2022).
7. Shen, Z.; Wang, L.; Deng, T. Digital Twin: What It Is, Why Do It, Related Challenges and Research Opportunities for Operations Research. 2 February 2021, p. 53. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3777695 (accessed on 6 March 2022).
8. Kuehn, W. Digital twins for decision making in complex production and logistic enterprises. *Int. J. Des. Nat. Ecodynamics* **2018**, *13*, 260–271. [[CrossRef](#)]
9. Kuliaev, V.; Atmojo, U.D.; Erla, S.S.; Blech, J.O.; Vyatkin, V. Towards Product Centric Manufacturing: From Digital Twins to Product Assembly. In Proceedings of the 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), Espoo, Finland, 22–25 July 2019; pp. 164–171. [[CrossRef](#)]
10. Bao, J.; Guo, D.; Li, J.; Zhang, J. The modelling and operations for the digital twin in the context of manufacturing. *Enterp. Inf. Syst.* **2018**, *13*, 534–556. [[CrossRef](#)]
11. Kulkarni, V.; Barat, S.; Clark, T. Towards Adaptive Enterprises Using Digital Twins. In Proceedings of the 2019 Winter Simulation Conference (WSC), National Harbor, MD, USA, 8–11 December 2019; pp. 60–74. [[CrossRef](#)]
12. Bárkányi, Á.; Chován, T.; Németh, S.; Abonyi, J. Modelling for Digital Twins—Potential Role of Surrogate Models. *Processes* **2021**, *9*, 476. [[CrossRef](#)]
13. Yildiz, E.; Möller, C.; Bilberg, A. Demonstration and evaluation of a digital twin-based virtual factory. *Int. J. Adv. Manuf. Technol.* **2021**, *114*, 185–203. [[CrossRef](#)]
14. Niati, A.; Selma, C.; Tamzalit, D.; Bruneliere, H.; Mebarki, N.; Cardin, O. Towards a Digital Twin for Cyber-Physical Production Systems: A Multi-Paradigm Modeling Approach in the Postal Industry. In Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings (MODELS '20 Companion), Virtual Conference, Canada, 12–23 October 2020; pp. 1–7.
15. Leitão, P.; Colombo, A.W.D.; Karnouskos, S. Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Comput. Ind.* **2016**, *81*, 11–25. [[CrossRef](#)]

16. Karnouskos, S.; Leitao, P.; Ribeiro, L.; Colombo, A.W. 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. [\[CrossRef\]](#)
17. Lazarev, A.; Pravdivets, N.; Werner, F. On the Dual and Inverse Problems of Scheduling Jobs to Minimize the Maximum Penalty. *Mathematics* **2020**, *8*, 1131. [\[CrossRef\]](#)
18. Alkhabbas, F.; Spalazzese, R.; Davidsson, P. An Agent-Based Approach to Realize Emergent Configurations in the Internet of Things. *Electronics* **2020**, *9*, 1347. [\[CrossRef\]](#)
19. Hrabia, C.-E.; Lützenberger, M.; Albayrak, S. Towards adaptive multi-robot systems: Self-organization and self-adaptation. *Knowl. Eng. Rev.* **2018**, *33*, e16. [\[CrossRef\]](#)
20. Chopra, A.K.; Singh, M.P. An Evaluation of Communication Protocol Languages for Engineering Multiagent Systems. *J. Artif. Intell. Res.* **2020**, *69*, 1351–1393. [\[CrossRef\]](#)
21. Dalpiaz, F.; Chopra, A.K.; Giorgini, P.; Mylopoulos, J. Adaptation in Open Systems: Giving Interaction Its Rightful Place. In *Conceptual Modeling—ER 2010/LNCS 6412*; Springer: Berlin/Heidelberg, 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. Bures, T.; Gerostathopoulos, I.; Hnetyinka, 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.
24. Gascuña, J.M.; Navarro, E.; Fernández-Caballero, A. Model-driven engineering techniques for the development of multi-agent systems. *Eng. Appl. Artif. Intell.* **2012**, *25*, 159–173. [\[CrossRef\]](#)
25. García-Sánchez, F.; Valencia-García, R.; Martínez-Béjar, R.; Breis, J.T.F. An ontology, intelligent agent-based framework for the provision of semantic web services. *Expert Syst. Appl.* **2009**, *36*, 3167–3187. [\[CrossRef\]](#)
26. Van der Valk, H.; Haße, H.; Möller, F.; Otto, B. Archetypes of Digital Twins. *Bus. Inf. Syst. Eng.* **2021**. [\[CrossRef\]](#)
27. Eramo, R.; Bordeleau, F.; Combemale, B.; van Den Brand, M.; Wimmer, M.; Wortmann, A. *Conceptualizing Digital Twins, hal-03466396, Version 1*; IEEE Software; Institute of Electrical and Electronics Engineers: Piscataway, NJ, USA, 2021; pp. 1–7. Available online: <https://hal.inria.fr/hal-03466396> (accessed on 3 March 2022).
28. Gruber, T. Toward principles for the design of ontologies used for knowledge sharing? *Int. J. Hum. Comput. Stud.* **1995**, *43*, 907–928. [\[CrossRef\]](#)
29. Lemaignan, S.; Siadat, A.; Dantan, J.Y.; Semenenko, A. MASON: A proposal for an ontology of manufacturing domain. In *Proceedings of the IEEE Workshop on Distributed Intelligent Systems—Collective Intelligence and Its Applications (DIS 2006)*, Prague, Czech Republic, 15–16 June 2006; pp. 195–200.
30. Usman, Z.; Young, R.I.M.; Chungoora, N.; Palmer, C.; Case, K.; Harding, J. A Manufacturing Core Concepts Ontology for Product Lifecycle Interoperability. In *Proceedings of the International IFIP Working Conference on Enterprise Interoperability IWEI 2011*, Stockholm, Sweden, 22–23 March 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 5–18.
31. Minhas, S.U.H.; Berger, U. Ontology Based Environmental Knowledge Management—A System to Support Decisions in Manufacturing Planning. In *Proceedings of the 6th International Conference on Knowledge Engineering and Ontology Development (KEOD)*, Rome, Italy, 21–24 October 2014; pp. 397–404.
32. Sormaz, D.; Sarkar, A. SIMPM—Upper-level ontology for manufacturing process plan net-work generation. *Robot. Comput. Integr. Manuf.* **2019**, *55*, 183–198. [\[CrossRef\]](#)
33. Järvenpää, E.; Siltala, N.; Hylli, O.; Lanz, M. The development of an ontology for describing the capabilities of manufacturing resources. *J. Intell. Manuf.* **2018**, *30*, 959–978. [\[CrossRef\]](#)
34. 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. [\[CrossRef\]](#)
35. 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.
36. Lu, Q.; Xie, X.; Heaton, J.; Parlikad, A.K.; Schooling, J. From BIM towards digital twin: Strategy and future development for smart asset management. In *Proceedings of the International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing*, Valencia, Spain, 3–4 October 2019; Springer: Cham, Switzerland, 2019; pp. 392–404.
37. 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. [\[CrossRef\]](#)
38. 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. [\[CrossRef\]](#)
39. 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. [\[CrossRef\]](#)
40. Rzevski, G.; Skobelev, P. *Managing Complexity*, 1st ed.; WIT Press: London, UK; Boston, MA, USA, 2014; p. 216.
41. Skobelev, P. Towards Autonomous AI Systems for Resource Management: Applications in Industry and Lessons Learned. In *Proceedings of the 16th International Conference on Practical Applications of Agents and Multiagent Systems (PAAMS 2018)*, Toledo, Spain, 20–22 June 2018; LNAI 10978. pp. 12–25.

42. Van Brussel, H.; Wyns, J.; Valckenaers, P.; Bongaerts, L.; Peeters, P. Reference Architecture for Holonic Manufacturing Systems: PROSA. *Comput. Ind.* **1998**, *37*, 255–274. [[CrossRef](#)]
43. Skobelev, P.; Zhilyaev, A.; Larukhin, V.B.; Grachev, S.; Simonova, E.V. Ontology-based open multi-agent systems for adaptive resource management. In Proceedings of the 12th International Conference on Agents and Artificial Intelligence (ICAART 2020), Valetta, Malta, 22–24 February 2020; SciTePress: Setúbal, Portugal, 2020; Volume 1, pp. 127–135.
44. Shoham, Y.; Leyton-Brown, K. *Multi-Agent Systems: Algorithmic, Game Theoretic and Logical Foundations*; Cambridge University Press: Cambridge, UK, 2009. Available online: <http://www.masfoundations.org> (accessed on 8 March 2022).
45. 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 8 March 2020).
46. Grachev, S.P.; Zhilyaev, A.A.; Laryukhin, V.B.; Novichkov, D.E.; Galuzin, V.A.; Simonova, E.V.; Maiyrov, I.V.; Skobelev, P.O. Methods and Tools for Developing Intelligent Systems for Solving Complex Real-Time Adaptive Resource Management Problems. *Autom. Remote Control* **2021**, *82*, 1857–1885. [[CrossRef](#)]