International Journal of Computer Integrated Manufacturing

ISSN: 0951-192X (Print) 1362-3052 (Online) Journal homepage: https://www.tandfonline.com/loi/tcim20

Trust-based resource sharing mechanism in distributed manufacturing

Ádám Szaller, Péter Egri & Botond Kádár

INTERNATIONAL JOURNAL OF COMPUTER INTEGRATED MANUFACTURING
2020, VOL. 33, NO. 1, 1–21
https://doi.org/10.1080/0951192X.2019.1699257

ARTICLE

Trust-based resource sharing mechanism in distributed manufacturing

Ádám Szaller[a,b], Péter Egri[a] and Botond Kádár[a]

[a]Centre of Excellence in Production Informatics and Control, Institute for Computer Science and Control, Budapest, Hungary; [b]Department of Manufacturing Science and Engineering, Budapest University of Technology and Economics, Budapest, Hungary

ABSTRACT
Manufacturing has moved from local towards global economy in the past decades enabling new paradigms to come into practice, such as distributed manufacturing which aims at increasing companies' flexibility with a decentralized system consisting of autonomous entities. In the paper, a distributed collaboration framework of manufacturing agents is introduced, where the members with resource shortages can request resources from others, divide requests among multiple agents, reorganize their production to be able to complete a request and cancel an undertaken task if needed. In a collaboration framework, it is essential to have a commitment to the promises: if participants cannot count on these commitments, the framework's efficiency may decrease. So as to motivate agents to keep their promises and to enable differentiating between reliable and non-reliable partners, here agents consider trustfulness during the selection from resource offering agents' proposals and make decisions considering subjective trust and public reputation values, which are computed based on successfulness of task performing and meeting due dates. In the paper, the impact of the proposed mechanism is investigated with multi-agent simulation. It is shown that considering trustfulness improves the overall system performance; and the improvement depends on the number of participants and the federation's load.

ARTICLE HISTORY
Received 21 May 2019
Accepted 26 November 2019

KEYWORDS
Distributed manufacturing control; collaborative manufacturing; distributed multi-agent systems; simulation; resource sharing; trust

1. Introduction

The phases of globalization have changed the existing paradigms of manufacturing. Although traditionally centralized approaches had several advantages, decreased transaction costs and opening of global markets moved the industry towards distributed production networks. (Lanza et al. 2019; Matt, Rauch, and Dallasega 2015). Demand fluctuations of today's volatile markets are also difficult to cope with, even when intelligent information technologies are applied in highly automated and connected smart factories corresponding to the Industry 4.0 paradigm (Váncza et al. 2011; Mourtzis 2011; Lanza, Peters, and Herrmann 2012; Hohmann and Posselt 2018; Pei et al. 2019). Rossit, Tohmé, and Frutos (2019) mention, these technologies change the way production planning and scheduling are carried out as well.

The producer–consumer relationships in production networks are also changing, which gives room for increased cooperation in order to cope with such problems (Kaihara et al. 2018). As Becker and Stern (2016) state, horizontal and vertical cooperation are distinguished among enterprises: if they are situated on different levels of a supply chain, the cooperation is called vertical, if they are at the same level of value-creation, it is a horizontal cooperation. In the past few years, several different manufacturing paradigms – which all build on increased cooperation between participants – were investigated by researchers. One of them is *cloud manufacturing (CMfg)* that transforms manufacturing resources and capabilities into manufacturing services, which can be managed and operated in an intelligent and unified way to enable the full sharing and circulating of manufacturing resources and manufacturing capabilities. A CMfg system includes a core support (knowledge), import and export of resources, and three user types: service providers, service users, and cloud providers (Zhang et al. 2014). *Sharing economy* is defined by Ter Huurne et al. (2017) as 'an economic model based on sharing underutilised assets between peers without the transfer of ownership, ranging from spaces, to skills, to stuff, for monetary or non-monetary benefits via an online mediated platform'. Another widely studied paradigm is *distributed manufacturing*, which aims at increasing the enterprises' flexibility and agility with a decentralized manufacturing system consisting of autonomous entities. As

CONTACT Ádám Szaller adamszaller@sztaki.hu Kende str. 13-17, Budapest H-1111, Hungary

manufacturing is moving from a local towards a global and competitive economy, fast and dynamic response to the customer demands is not possible with a system built on rigid, hierarchical and centralized control architectures – which, in addition, may shut down due to a single failure. In distributed manufacturing (and in general, in a distributed control system), each participant is intelligent, cooperative, has its own objectives, skills and knowledge – however, none of them has a global view of the system (Leitao 2009). To increase flexibility, participants can share resources with each other, which also requires information sharing. Sharing resources in a distributed system in a collaborative way is called *crowdsourced manufacturing* by Kádár et al. (2018). In such a system, companies or producer entities cooperate within a brokering *federation* in order to reach higher and more competitive service levels. Producers, usually named factory agents, share their manufacturing assets on the basis of their actual and expected orders or available extra capacities.

In case of these new manufacturing paradigms, one wonders how can they be implemented in the industrial practice, and what is the motivation for organizations to share information and resources in a competitive market situation. For example, the Swiss Virtuellefabrik (Swiss Virtuellefabrik 2019) is a collaborative production framework consisting of small enterprises, with basically complementary resources and competences. They mainly focus on manufacturing unique products and prototypes – which cannot be done without specific capabilities. The orders are placed, and the parts of the task are distributed between the manufacturers through special brokers. The goal of the framework is to complete orders together, complementing each other's competences and resources. The participants of the framework could not utilize their resources in an efficient way without working together.

Kaihara et al. (2017) also indicate that because of frequent demand changes, for Build-to-Order (BTO) companies it is difficult to reach a high machine utilization level, since to keep the due dates, such companies usually have excess production capacities. The solution could be to share resources with each other in a crowdsourced way – however, some risk exists that manufacturing costs might become higher than usual because of additional transportation and subcontracting costs.

When cooperating with each other, organizations can also offer resources to each other as services – this has been already implemented to the industry in case of, e.g. 3D printing and laser cutting companies, to whom a specific task of a production or manufacturing process can be outsourced. Váncza et al. (2011) mention a machine service network named 3DWorknet, in which the focus is on fabrication in standardized production plants connected to a network – which aim is to improve the efficiency of the process chain in the manufacturing industry. An example of this type of plant is Shapeways Portal (Shapeways Portal 2019), where customers either have the opportunity to order existing 3D models from a library or upload their own models to be printed. Other companies who are offering manufacturing services are Fictiv (Fictiv Online Manufacturing Platform 2019) and Plethora (Plethora CNC machining on demand 2019). These companies operate online platforms where the customers can upload CAD models about the product they want to manufacture, and the companies produce them in as short time as 1 to 3 days.

1.1. Resource sharing models in the literature

Sharing resources in production structures has the potential to increase flexibility and scalability of manufacturing systems – as mentioned by Freitag, Becker, and Duffie (2015), where a resource sharing model is introduced, and different scenarios are tested with increasing degree of information exchange between the participants. Shi et al. (2007) investigate resource sharing by applying grid technologies for resource modelling. The distributed and dynamic allocation of resource capacities have long been proposed for decentralised scheduling applying an agent-based architecture (Kumara, Lee, and Chatterjee 2002). More recently, the capacity allocation problem has been studied on the network level as well (Scholz-Reiter et al. 2011).

An important precondition of resource sharing is the capability of checking planned resource utilizations and determining shortage or surplus of capacities in advance. Most Enterprise Resource Planning (ERP) systems include a Capable-to-Promise (CTP) module, which can determine whether there are enough materials and capacities for satisfying incoming customer orders (Capable-to-Promise Systems

2019). While the advantage of the CTP is that it is already available at most manufacturing sites, more sophisticated scheduling or simulation systems are preferred not only for availability checking but also for modifying the production plans and schedules according to the new customer orders (Kaihara et al. 2017; Kádár et al. 2018). Chen et al. (2008) present a distributed access control architecture for collaborating dynamic virtual enterprises – that may both compete and cooperate with each other – to solve problems in distributed authorization management and security access control across organizations. Liu et al. (2015) introduce a resource service sharing model in cloud manufacturing, which is based on the Gale-Shapley algorithm, and analyse it in the context of fluctuating resource service supply and demand. When collaborating with each other, the participants often have to make decisions about who to work with. Arrais-Castro et al. (2018) propose a model, which uses a dynamic decision approach for supplier and business partner evaluation in a collaborative network – here, software agents autonomously capture business opportunities, select business partners, and award associated orders.

In a system where participants share resources and cooperate with each other, it is also important for them to have a strong commitment to their own goals and plans. Otherwise, they would bring nothing to completion. They somehow have to be encouraged to keep their promises, because in a cooperative framework trusting in each other's promises is one of the main pillars of the system (Váncza and Márkus 2000). This paper indicates these essential preconditions of collaboration as *trust* and *reputation*. However, these are less exact terms, which are more difficult to define, since they are based on a complex belief of dependability, competence and integrity. Hence, it is beneficial to briefly review here the general characterization of trust and reputation systems (TRSs).

1.2. Trust and reputation systems

Extensive reviews of computational TRSs used in multi-agent systems are presented by Sabater and Sierra (2005) and Pinyol and Sabater (2013). These studies introduce a classification of the models according to a number of dimensions. In general, two paradigm types are distinguished: *cognitive*, where trust and reputation are built on beliefs and their degrees, and *numerical*, where the values are calculated from utility functions and numerical aggregation of past interactions. The models consider different information sources to calculate trust and reputation. The agents can make decisions based on *direct experiences* (direct interactions and direct observations), *indirect information*, which is gathered from other agents, *sociological information*, which is based on the analysis of social relations among the agents, and *prejudice* (assigning properties to an individual, based on signs that identify the individual as a member of a given group). The type of information can also be *discrete-valued* (e.g. the agent met the due date or not) and *continuous-valued* (lateness in the due date). Another important aspect is visibility: values can be *public* (visible for all the observers) or *subjective* (assessed by each individual). It is also an essential question whether the models take the reliability of measures into consideration: is trust and reputation single-valued without any other information or do they contain other elements, e.g. number of experiments, reliability of witnesses or the age of a specific information.

As mentioned, there is no widely accepted definition for trust and reputation, but in most practical TRSs, trust means a subjective value that is based on direct experiences, and reputation is a public value, which includes indirect information. For practical purposes, simple numerical characterizations of trust and reputation can be applied, e.g. the average order fill-rate based on historic interactions – as presented by Hou et al. (2018), where the effect of trust in supply chains is investigated. Cheikhrouhou, Pouly, and Madinabeitia (2013) distinguish between five trust categories (competence, contractual, relational, indirect and negative), and investigate their impacts on information exchange processes in vertical collaborative networked organisations. Li, Fan, and Xitong (2011) also state that trust plays an important role in the selection of business services. Chang et al. (2014) propose a multi-criteria variable weights decision-making approach based on trust and reputation in supply chains. They put more emphasis on the detailed TRS, consider direct and indirect values and apply a time decay function for historical trust and reputation values as well, but ignore the resource constraints at the suppliers. Yang et al. (2019) present a service satisfaction-based trust evaluation model for cloud manufacturing, where the direct satisfaction, the friend recommendation satisfaction and the platform satisfaction were integrated into the comprehensive trust. This model also applies a time decay function and corrects trust values by using the service satisfaction volatility. Yan, Cheng, and Tao (2016) present a detailed TRS in Cloud Manufacturing. Here, direct, indirect, and third-party trust (which means relying on the opinion of independent and qualified third parties) are taken into consideration, and a time decay function is applied on historical transaction data. The model focuses on the trust evaluation model and takes into account several important aspects in connection with trust, but also ignores resource constraints. Nevertheless, the amount of available resources is an important aspect when investigating systems where the participants (e.g. companies) share resources with each other. A reliable participant could become overloaded and, as a consequence, other companies might choose a less reliable partner with free capacities instead of the reliable one which has no available resources.

For cooperating organizations, it is essential to be honest with each other and to have a strong commitment to the promises. With taking trust and reputation into account in decision-making, companies could be incited to keep their promises, e.g. complete an undertaken order in spite of noticing a more profitable option for using free capacities. They also can be forced not to bias information and to meet the task due dates because otherwise they would worsen their own situation (after receiving a bad rating, they are less likely to win new tasks). Making decisions based on trust and reputation also enables to differentiate between partners who are reliable, and who are not. Such a framework is driven by the promises and commitments for the future, given by the participants. The main pillar of the framework is that one can believe the other's promises: if participants cannot count on these commitments, and they are not incited to keep the promises, the framework of cooperation is violated, and the efficiency of the distributed manufacturing system can but decrease.

1.3. Preliminaries

From framework point of view, the paper presents the extension of the former distributed collaboration framework introduced by Kádár et al. (2018), which was developed with the aim of facilitating the cooperation of manufacturing companies. Here, all companies (modelled with agents) are able to offer their free resources and send requests when having resource shortages, as well. A Collaboration Platform (CP) receives the offers and requests to match them (if possible). The agents are members of a federation, which is a group of agents, with the CP in the centre. The advantage of the approach is that the capacity requesting agent receives an instantaneous reply for its request and thus the decision process is not delayed. In contrast, it disregards the potential flexibility of the production at the other agents, which may decrease the efficiency. Besides, this protocol generates high communication load as it records all the free capacities, requests and offers in one central system, and updates them continuously. In addition, the study focused only on the case when the agents received prompt answer from the platform: if there is no matching offer at the moment when a certain request arrives, the matching fails. It is also not realistic for a production facility to share all the capacity information about itself (except that all the participants of the system belong to the same company, as different production sites). In reality, companies try to share as little information with the others as possible. The mentioned paper does not take the trustfulness of agents into consideration and does not reward or penalize the federation members on the basis of how they kept their promises regarding task fulfilment.

In this paper, the authors introduce an abstract agent-based model, whose purpose is to investigate the impact of considering trust and reputation in decision-making while sharing resources in a distributed manufacturing system. Trust and reputation between participants are based on the task fulfilment successfulness and finishing time (whether the company met the promised due date). The present model includes both the flexibility of the agents allowing them to divide requests with the aim of finding an appropriate offer, to reorganize their production in order to fulfil a resource request, and the evaluation of their trustfulness as well – resulting in a more adaptive model. It is also possible for an agent to cancel a task it promised to complete – in this case, its trust and reputation will decrease. After a detailed description of the formal model, the effect of considering trust and reputation in decision-making is evaluated using multi-agent simulation. It is shown that in a distributed manufacturing system consisting of cooperative agents who are sharing resources with each other, considering trust and reputation improve the system's overall performance. In addition, the improvement increases as the number of participants grows in such a system. Experiments also have shown that if the load of the federation increases, taking trustfulness into account has less impact.

2. Formalized model

2.1. Basic concepts

In the model description, the following notions will be used:

- *Agent*: represents a company which has a certain amount of different types of resources. It can communicate with other agents, offer its resources and send requests to other agents with the aim of asking for additional resources, if it is necessary. On the basis of its decision mechanism, it can choose the best from the incoming offers.
- *Federation*: a group of agents. Agents are allowed to enter or exit the federation at any time: the entry condition is to accept the interaction protocol. Collaboration is only possible between federation members.
- *Federation Centre (FC)*: manages entries and exits from the federation, updates the list of federation members, and calculates reputation values for each member.
- *Task*: a specific production process that has to be performed by the agents. A task is determined by its resource requirements:
    - One specific *resource type* that is necessary to perform it (e.g. drilling machine).
    - *Amount of required resources* (continuous-valued parameter). Each company has a given amount of resources from a specific type; the mentioned amount cannot be used for other purposes in the task processing interval (defined below).
    - *Earliest start time* and *due date* determine a *processing interval* where resources with the given amount are to be used.
- *Resource load*: in case of a specific task, the amount of required resources multiplied by the length of its processing interval.
- *Order*: contains one simple task. Federation members receive a stream of orders from outside the federation.

2.2. Model structure

In the presented model, the authors consider two aspects.

(1) The goal of a federation member: similar to real companies that are trying to generate revenue from completing jobs, in the presented model agents are motivated to perform as many tasks as possible and utilize their resources as much as possible – however, financial aspects are not modelled in detail.
(2) The goal of the whole federation: to maximize service level for outside customers and perform the undertaken tasks on time.

In this paper, the focus is on the performance of the federation and its collaborating members, who make decisions on the basis of the other federation members' trust and reputation values. Agents have an internal decision-making mechanism for deciding which order (received from outside the federation) to accept or reject – however, in the model, the authors only deal with accepted orders. Here, each federation member undertakes some orders – even lacking sufficient capacities that are required to the certain order, relying on the strength of the federation, and assuming that the member(s) of the federation will help to complete the order. There can be a task which the agent is not able to perform either because of resource shortage or even the lack of specific resource. In this case, that task has to be outsourced – enabling the requesting agent to complete the order. In Figure 1, one can see the federation with company agent members. They receive an order stream consisting of a series of tasks from outside the federation (light blue arrows), collaborate with each other by sending and receiving requests, offers and messages (black arrows), and communicate with the FC (red arrows).

Since the main goal of this paper is to investigate the effect of considering trust and reputation in decision-making, and for simplicity reasons, the resource amount is considered to be continuous valued in the

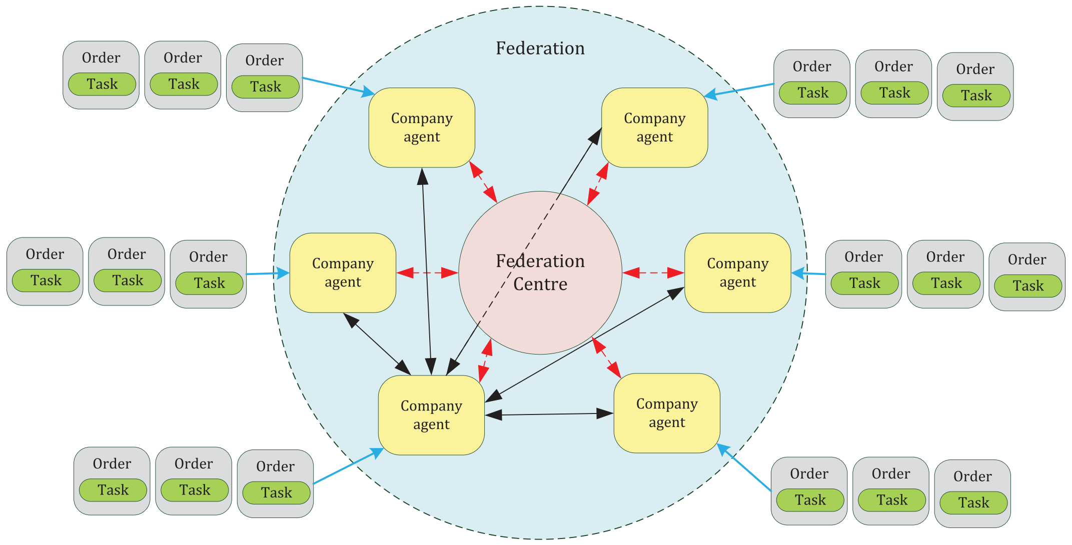

Figure 1. Federation of agents with incoming orders.

presented model. Main parts of the communication protocol are the following:

- *Request*: a message sent by an agent which has resource shortages to all the other agents with the aim of asking for additional resources in connection with a task. A request consists of the resource requirements (type, amount, earliest start time and due date) of a specific task. The actual list of federation members is provided by the Federation Centre.
- *Capability check*: when an agent receives an order from outside the federation or a request from another federation member, it checks whether it has the resource requirements to perform it. If the agent is able to complete the task, the result of the capability check is true, otherwise, it is false.
- *Requesting agent*: in case of an incoming order, if the result of the capability check is false, the agent sends requests to all the other federation members – in this case, this agent is called 'requesting agent'.
- *Offering agent*: in case of an incoming request, if the result of the capability check is true, the agent which received the request sends an offer to the requesting agent – in this case, the agent which sent the offer is called 'offering agent'. Agents could request and offer resources related to different tasks at the same time: the notion depends on the role of the agent in the specific interaction.
- *Winner agent*: the agent that has sent the best offer and has been chosen by the requesting agent to complete a specific task.

2.3. *Trust and reputation definition*

According to the classification mentioned in Section 1.2, the trust model considered in this paper is a numerical one, with continuous-valued information. When an agent undertakes a task, it makes two promises:

(1) to complete the task, and
(2) to complete it on time.

Trust and reputation are formed by the fulfilment of these two promises, which are related to the reliability and trustworthiness of the agents but calculated in different ways (presented in subsection 2.9). In the above distributed production network, the top priority is maximizing service level of the federation, thus, if the task is completed, trust and reputation values are calculated on the basis of the lateness in the outsourced task completion times. Whenever an agent promised to complete a specific task, but later refuses it, the trust and reputation of the agent decrease. Hence, trust and reputation are defined as follows (more details about the calculation method will be provided in the model description):

- *Trust*: internal, subjective value. Each agent associates a trust value to each of the other agents. It is calculated by the agents themselves, on the basis of direct interactions with another specific agent taking successfulness of the interaction and task lateness into consideration. Trust can be considered as a subjective opinion about everyone else in the federation. It has a value between 0 (lowest trust) and 100 (highest trust).
- *Reputation*: public value, assigned to each agent. It is calculated by the FC but formed by the interactions with all the other agents taking successfulness of the interaction and task lateness into consideration. It can be considered as specific rating that each agent can see and influence and qualifies the agent for the others. It has a value between 0 (lowest reputation) and 100 (highest reputation), too.

Therefore, based on the notions in Sabater and Sierra (2005) and Pinyol and Sabater (2013), direct and indirect information are also considered in the model, which applies subjective and public values as well. As trust and reputation values are changing over time, former values are weighted by an exponential function when calculating new rates. In the current stage of research, agents are completely honest; they are not allowed to manipulate the system with communicating distorted information, e.g. devaluate a partner for selfish reasons in spite of its high reliability.

2.4. Agent interaction

The flowchart of the agent interaction applied in the model is presented in Figure 2. When an agent receives an order from outside the federation (1), it performs the capability check. If the result is true, the agent schedules the task for itself based on the earliest task start time and due date – and performs it between these two time points. If the result is false, the agent checks the federation member list (updated after each entry or exit by the FC) and sends requests to all the other agents of the federation immediately (2). It is necessary to send the request to all the federation members because agents do not have any information about each other's resource types or amounts. After receiving an order, agents perform the capability check on their own production plan and send an offer to the requesting agent if the result is true – otherwise, send a reject message about offering their resources. Offer or reject message sending occurs in a time that is determined with a uniform distribution between 0 and 1 model time unit for each agent. The requesting agent expects some kind of answer from each of the other agents in one model time unit – an offer is technically a feedback that the offering agent is able to complete the specific request.

If the requesting agent does not receive any offer from the other agents, it sends out the request again, and in parallel asks all the other agents to try to reorganize their production with the aim of completing the specific task (3). They check their production and free resources again and if it is possible to complete the task after reorganizing (this process lasts for another model time unit), they send back an offer or a reject message. If there are still no offers, the requesting agent divides the task to equal parts, sends out its parts separately as requests, and waits another time unit for offers (4). If the requesting agent does not receive offers for some of the parts, marks them as 'failed' (5). If there is at least one offer after step (2), (3) or (4), the requesting agent chooses the best (or the only) offer and assigns the task to the winner (offer evaluation will be described later in this section). In Figure 2 the frames in blue-dashed lines are the same steps that the agent performs when receiving a request at different phases of the interaction.

After the task is finished or cancelled, the requesting agent updates the winner agent's (subjective) trust value, and the FC updates the winner's (public) reputation value depending on whether the agent completed the task or cancelled it (6a and 6b). If the task is completed, trust and reputation values will change according to the lateness in the due dates (detailed in Section 2.9). If the task is cancelled, trust and reputation values will be recalculated by assigning a zero value to this unsuccessful interaction. In this case, the requesting agent does not try to find a new offer for this task and marks the task as failed. If a (part of a) task is marked as failed, the requesting agent does not try to send it out again.

2.5. Capability check

For the figures and equations in the following sections, Table 1 contains the description of the variables.

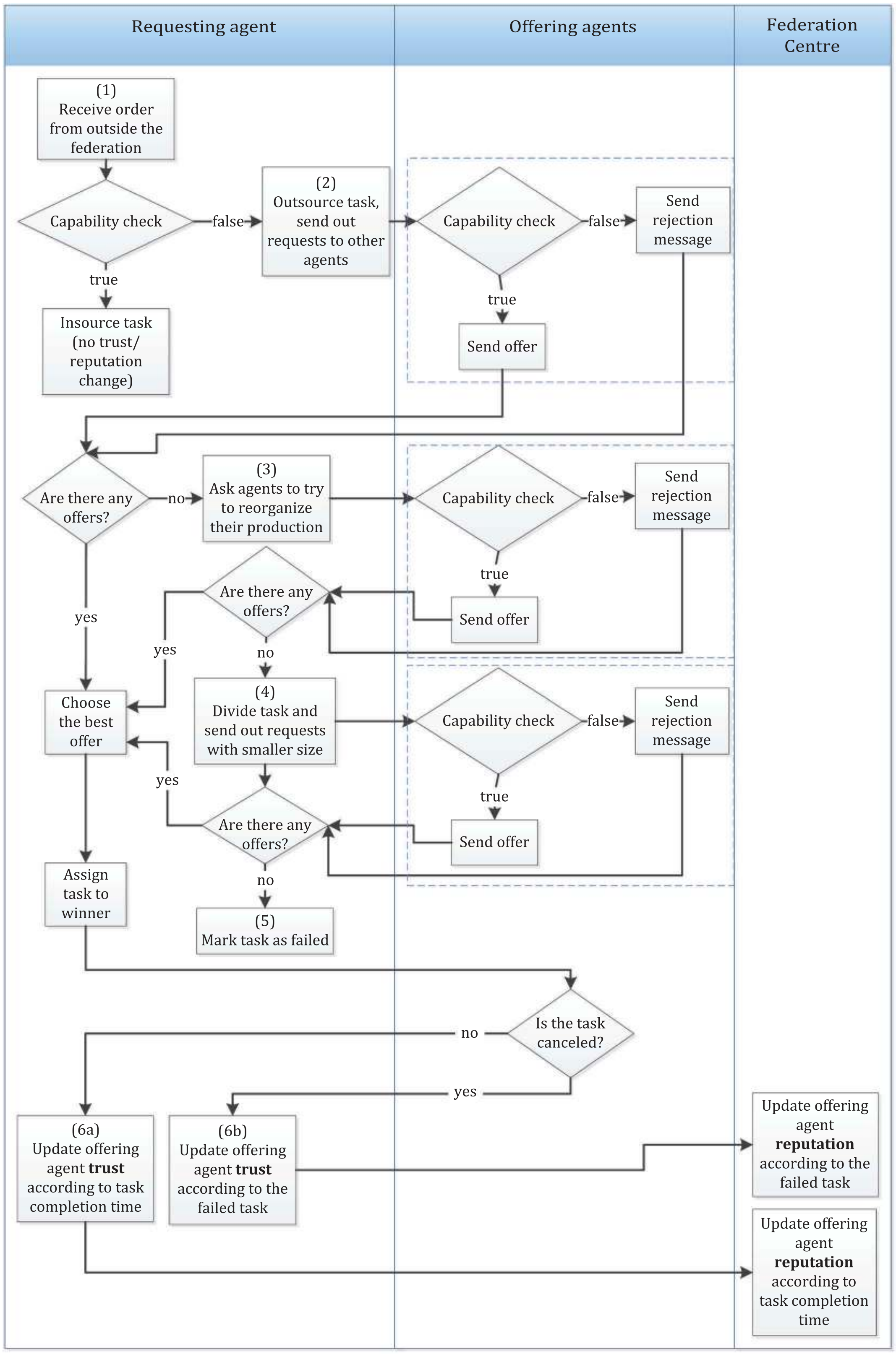
Requesting agent
Offering agents
Federation Centre
(1) Receive order from outside the federation
Capability check
false
true
(2) Outsource task, send out requests to other agents
Insource task (no trust/ reputation change)
Capability check
false
Send rejection message
true
Send offer
Are there any offers?
no
yes
(3) Ask agents to try to reorganize their production
Capability check
false
Send rejection message
true
Send offer
Are there any offers?
yes
no
Choose the best offer
(4) Divide task and send out requests with smaller size
Capability check
false
Send rejection message
true
Send offer
Are there any offers?
yes
no
Assign task to winner
(5) Mark task as failed
Is the task canceled?
no
yes
(6a) Update offering agent trust according to task completion time
(6b) Update offering agent trust according to the failed task
Update offering agent reputation according to the failed task
Update offering agent reputation according to task completion time

Figure 2. Swim lane of the distributed resource sharing model.

When $agent_n$ receives a new task ($task_{new}$) with t_{new} processing interval and r_{new} resource amount, it tries to insert the task into its production plan (Figure 3 and Equation (1)). The agent calculates the cumulative planned load of the specific resource type during the processing interval of the new task by

Table 1. Variables in the model description.

Variable	Description
t_i	processing interval of $task_i$
t_{new}	processing interval of a new, incoming task (that the certain agent tries to fit in to its production plan)
r_i	required amount of resources for $task_i$
r_{new}	required amount of resources for the new, incoming task
$r_{max,n}$	maximum amount of resources from a specific resource type for $agent_n$
$o_{i,j}$	overlapping time interval of $task_j$ with $task_i$
p_n	number of already planned tasks for $agent_n$
$task_{k,i}$	kth part of $task_i$ (after dividing)
$r_{inc,i}$	increased amount of required resources for $task_i$ (due to delayed start)
$t_{delay,i}$	difference between the real start time and the earliest start time for $task_i$ (in case of delayed start)
L_i	difference between the promised finishing time and the real completion time of $task_i$ (lateness)
$trust_i^{m,n}$	trust value in connection with $task_i$, which was requested by $agent_m$ and won by $agent_n$
$rep_i^{m,n}$	reputation value in connection with $task_i$, which was requested by $agent_m$ and won by $agent_n$
$trust_{prev}^{m,n}$	previous subjective trust value in case of a finished task (how much $agent_m$ trusts $agent_n$)
$trust_{new}^{m,n}$	new subjective trust value after a finished task (how much $agent_m$ trusts $agent_n$)
rep_{prev}^{n}	previous public reputation value of $agent_n$ in case of a finished task
rep_{new}^{n}	new public reputation value of $agent_n$ after a finished task

summarizing the load of all p_n already planned tasks and subtracting it from the possible maximum resource load ($r_{max,n} \cdot t_{new}$) that the agent can use in this interval from a specific resource type. If the difference is higher than the resource load of the new task, the agent is able to complete the request and sends an offer to the requesting agent.

$$r_{new} \cdot t_{new} \leq r_{max,n} \cdot t_{new} - \sum_{i=1}^{p_n} r_i o_{i,new} \tag{1}$$

In Figure 3, a simple example is shown, where $task_i$ and $task_j$ are already planned tasks with t_i and t_j processing intervals, and r_i and r_j resource amounts. The overlapping time intervals with t_{new} is $o_{i,new}$ and $o_{j,new}$. Here, the area of the hatched rectangle is equal to the possible maximum resource amount the agent can provide in the t_{new} interval. The agent subtracts the overlapping area of the two light blue rectangles (planned tasks) from the area of the hatched rectangle (maximum resource load), and if the difference is higher than the area of the green rectangle (new task), it sends an offer to the requesting agent. In this case, the offering agent has enough resources to perform $task_{new}$. In order to have a manageable model, setup time is included in the processing interval.

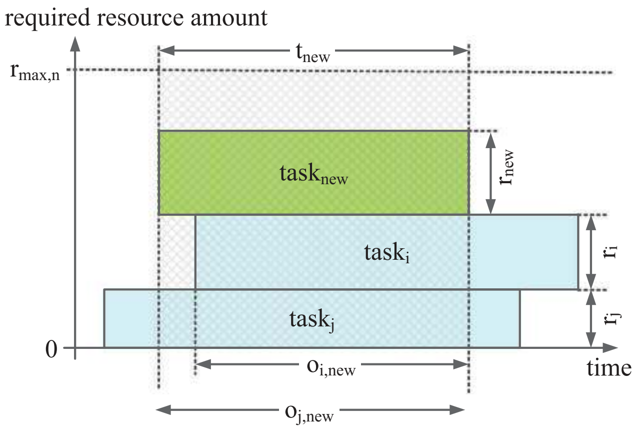

Figure 3. Calculation of available resources in case of a new task.

2.6. Changes in task parameters due to delayed start

In the presented model, the earliest start time of a task is at least one model time unit later than the time when the order (containing the task) was received by the requesting agent. In this case, the requesting agent certainly receives the offers or refuse messages related to a specific task before the earliest start time occurs. However, during further interactions, it may happen that by the time an appropriate offer is found, the execution of the task should have already started. It is assumed that the task can still be completed in this case, but since there is less time available to finish it, more resources are required in order not to change the due date. To finish a task, the required resource load (resource amount multiplied with the processing interval) is necessary – thus if there is less time available, more resources are needed. As shown in Equation (2), the product of the original required resource amount r_i and the original processing interval t_i is equal to the increased amount of resources $r_{inc,i}$ multiplied with the original processing interval t_i reduced by the delay (difference between the real task start time and the earliest start time; $t_{delay,i}$).

$$r_i \cdot t_i = r_{inc,i} \cdot (t_i - t_{delay}) \tag{2}$$

Figure 4 shows the same calculation, but in a visual way: here, $task_i$ is the original task, visualized with a light blue rectangle with $r_i \cdot t_i$ area. If performing a task does not start at the earliest start time, the area of the rectangle – i.e. the resource load of the task – should not change. The orange rectangle indicates the modified task – due to delayed start –, this has the same area as the light blue one.

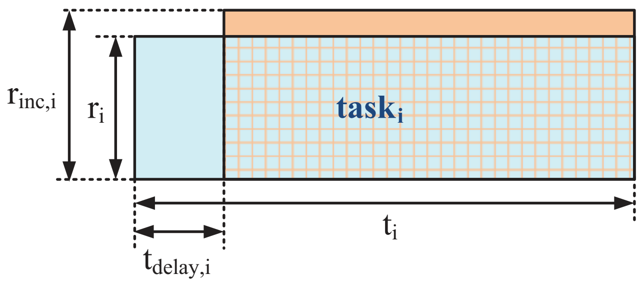

Figure 4. Increased required resource amount due to delayed start.

2.7. Reorganisation

When an agent is asked by a requesting agent to reorganize its production, it has the opportunity to use additional resources in the interval given by the earliest start time and the due date of a task that is planned to be completed. Companies can rearrange the scheduled tasks with the aim of completing a new task or to use additional resources (e.g. to call in more employees to the shop floor or ask them to work overtime). In case of reorganizing, when an *agent*$_n$ checks its free resources according to Equation (1), the $r_{max,n}$ maximum resource amount could be increased in the mentioned interval. The increase rate in the percentage of the original amount is an agent parameter called *flexibility*, this way modelling the usage of additional resources.

2.8. Dividing tasks

The tasks are assumed to be dividable to some parts: if asking the other agents to reorganize their production does not lead to new offers, the requesting agent divides the task to smaller, equal parts, and sends them out separately with the aim of finding offers for the smaller tasks. The number of the parts is a model parameter, and constant in all interactions. Dividing a task means that the processing interval remains the same, but smaller resource amount is necessary to complete one part, and the sum of the resource loads of the parts are equal with the original task resource load (Figure 5). In case of dividing, the requesting agent first computes the required (possible increased) resource amount according to Equation (2), then divides the task to a parametrizable number of parts, and finally sends out new requests with the divided requirements and possibly delayed earliest start times. In this case, there

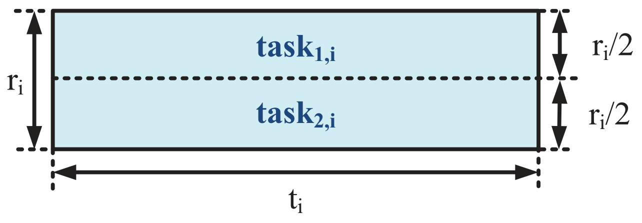

Figure 5. Dividing a task into two parts.

is a higher chance that separate agents have free resources to complete the smaller requests. In Figure 5 one can see a case when a task is divided into two equal parts ($task_{1,i}$ and $task_{2,i}$). Both has t_i processing interval and requires $r_i/2$ resources.

2.9. Computation of trust and reputation values

If the requesting agent receives more than one offer, it has to decide which one to select. When making this decision, it takes the weighted sum of three factors into consideration: the *unit price* (static agent feature), the *reputation*, and the *trust* value of the offering agent. Agents make decisions considering static and dynamic parameters as well, resulting in a more adaptive decision-making. Trust and reputation values are related to the successfulness of task completion and the lateness of the task completion times – in this way, agents rate each other based on their past performance and affect the system behaviour. Both trust and reputation values are changed after each finished task, but mean different things, as explained in Section 2.3.

Before evaluating an offer, the requesting agent always checks the reputation value of the offering agent through the FC, and the trust value calculated and stored by itself. When a task is finished, first, $trust_i^{m,n}$ and $rep_i^{m,n}$ – trust and reputation values in connection with $task_i$ which was requested by agent$_m$ and won by agent$_n$, and has a processing interval t_i – are calculated according to Equation (3). The trust and reputation values in connection with $task_i$ are the same, they will be the base of the change in the trust and reputation values of the winner agent, and they are influenced by the L_i lateness between the promised finishing time and the real completion time of $task_i$ (here the authors suppose that $t_i \geq L_i$ in order to trust and reputation values remain positive). The model contains an α penalty parameter to sanction lateness to a greater extent and increase the

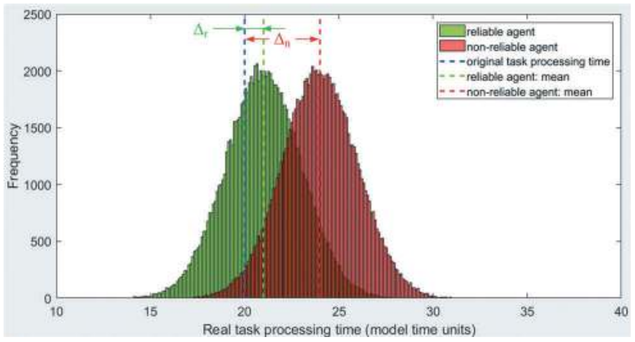

Figure 6. Difference between reliable and non-reliable agents.

difference between agents based on their performance. The value of α depends on the task finish time:

- if $L_i \leq 0$, $\alpha = 1$
- if $L_i \geq 0$, $0 < \alpha < 1$ (constant model parameter)
- if the task is cancelled, $\alpha = 0$.

Therefore, if the winner agent finishes the task earlier than the due date or right on time, it gets $trust_i^{m,n} = rep_i^{m,n} = 100$ value (it does not matter how much earlier the agent finished it). If a task is cancelled, $trust_i^{m,n}$ and $rep_i^{m,n}$ will be equal to 0.

$$trust_i^{m,n} = rep_i^{m,n} = \left(100 - \frac{L_i \cdot 100}{t_i}\right) \cdot \alpha \qquad (3)$$

As one can see in Figure 6, real processing intervals are determined by normal distributions – this causes the L_i lateness in the completion time of $task_i$. In the presented model, the mean of the distribution of the real processing interval is always higher than the original processing interval. The authors distinguish agents according to their service level as *reliable* and *non-reliable* agents: agents which belong to the second type are more likely to finish a certain task later than the due date, and it is expected that other agents are not willing to outsource tasks to them as frequently as to the reliable ones. However, it is difficult to mark a clear boundary between reliable and non-reliable agents; in the present model an agent is considered reliable if the difference between the mean of the real processing interval and the original processing interval is smaller or equal than 10% of the original processing interval, and non-reliable if it is higher than 10%. This difference is a static agent feature, and it has a big impact on trust and reputation values, and at the current stage of research, does not depend on the workload of an agent in the specific time point or other parameter. In Figure 6, Δ_r and Δ_n denote the mentioned time difference in case of reliable and non-reliable agents (the index r refers to the reliable, n to the non-reliable agents in this paper).

Task cancelling also strongly influences the trust and reputation values, thus indirectly the number of tasks the agent wins. Task cancelling can happen with a specific probability in case of each type of agent – this probability is denoted by x_r and x_n, and given in the percentage of all tasks requested by other federation members (e.g. $x_n = 20\%$, means that 20% of all the won tasks will be cancelled by non-reliable agents).

After the requesting agent calculated $trust_i^{m,n}$ and $rep_i^{m,n}$, the requesting agent changes its subjective trust value about the winner agent on the basis of Equation (4), sends $rep_i^{m,n}$ to the FC, and finally, the FC changes the winner agent's public reputation value according to Equation (5). In Equation (4), $trust_{new}^{m,n}$ means the new, and $trust_{prev}^{m,n}$ means the previous subjective trust value of the winner agent, which refers how much $agent_m$ trusts $agent_n$. In Equation (5), rep_{new}^{n} means the new, and rep_{prev}^{n} means the previous public reputation value of the winner agent. Here, there is only one index, because this value is connected only to the winner $agent_n$, as it is shaped by all interactions performed by this agent. Exponential smoothing with smoothing factors β (in case of trust) and γ (in case of reputation) were applied to assign exponentially decreasing weights over time to trust and reputation when calculating new values. For initial trust and reputation values, 80 is considered (on the 0–100 scale). Important to note that trust values are not symmetric: $trust^{m,n} \neq trust^{n,m}$.

$$trust_{new}^{m,n} = trust_i^{m,n} \cdot \beta + trust_{prev}^{m,n} \cdot (1 - \beta) \qquad (4)$$

$$rep_{new}^{n} = rep_i^{m,n} \cdot \gamma + rep_{prev}^{n} \cdot (1 - \gamma) \qquad (5)$$

3. Simulation experiments

In order to investigate the performance of the federation and the individual agents, some experiments were performed. For these investigations, a high-level multi-agent-based simulation model was built in AnyLogic (Borschev 2013). In most of the experiments, the resource sharing mechanism described above has been implemented with 10 company agents, with the aim of testing and validating the proposed protocol. As one can see in Table 2 six non-reliable (C01 to C06 marked with red background) and four reliable (C06 to C10 – marked with green background) company agents were considered. In case where not only 10 agents were investigated, the agents in Table 2 were duplicated as detailed in the description of the specific experiment. In the experiments, 20 different resource types existed; the types and amounts are also listed in Table 2 (for example, C01 company agent has 10 units form resource type 1). As one can see, two of the non-reliable agents (C01 and C02) has all the 12 resource types, the others have only 10 of them.

The model has several parameters, thus, for the easier understanding, Table 3 contains the varied parameters and Table 4 contains the parameters that were fixed in all experiments.

In Table 5 the performed experiments are introduced, with the bounds of the varied parameters, which are marked with a grey background. If there are no grey background cells in the column of an experiment, that means a specific scenario – detailed in the experiment description – was investigated. In case of most of the experiments, the simulation was run for 750 model time units: after this time period, the measured KPIs did not change significantly. In Experiment (8) the effect of an unexpected negative event is investigated, and here it was necessary to run the simulation two times longer than in the other cases to show the system changes.

In the experiments, when the authors compare the *normal* and the *advanced* model, normal means the model introduced in Section 2 *without computing trust and reputation or considering them in decision making*. Unit price was taken into account in all cases: when no trust or reputation values were considered (normal model), agents made decisions based only on this static feature. In the other cases, a weighted sum of unit price and trust/reputation values were calculated and associated to a certain offer during evaluation. In the experiments, these weights were equal. As one can see from Table 4, the unit prices are also equal for these experiments in order not to influence the difference between reliable and non-reliable agents. This way, when applying the normal model, the agents will send equal offers, thus, the first received offer will be the winner.

Table 2. Companies in the simulation experiments.

Resource type	Company									
	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10
1	10	10	10		10				10	10
2	12	12	12		12				12	12
3	10	10	10		10				10	10
4	12	12	12		12				12	12
5	10	10	10		10				10	10
6	8	8	8		8				8	8
7	9	9	9		9				9	9
8	10	10	10		10				10	10
9	11	11	11		11				11	11
10	12	12	12		12				12	12
11	10	10		10		10	10	10		
12	12	12		12		12	12	12		
13	10	10		10		10	10	10		
14	12	12		12		12	12	12		
15	10	10		10		10	10	10		
16	10	10		10		10	10	10		
17	12	12		12		12	12	12		
18	10	10		10		10	10	10		
19	12	12		12		12	12	12		
20	10	10		10		10	10	10		

Table 3. Varied parameters in the experiments.

Notation	Description	Unit
a_r	number of reliable agents	pcs
a_n	number of non-reliable agents	pcs
d	number of parts the agents can divide the requests into	pcs
f	agent flexibility	%
tr	the model includes trust (1) or not (0)	-
rep	the model includes reputation (1) or not (0)	-
β	smoothing factor for trust values	-
γ	smoothing factor for reputation values	-
t_{order}	incoming order time period (difference between two incoming orders to an agent)	model time unit

Table 4. Fixed parameters in the experiments.

Notation	Description	Value	Unit
Δ_r	difference between the mean of the real processing interval and the original processing interval, in case of *reliable* agents	5	%
Δ_n	difference between the mean of the real processing interval and the original processing interval, in case of *non-reliable* agents	20	%
α	penalty parameter in case of delayed tasks	0.8	-
x_r	task cancelling rate in case of *reliable* agents	2	%
x_n	task cancelling rate in case of *non-reliable* agents	20	%
t_{avg}	average interval size of the tasks received from outside the federation	40	model time unit
r_{avg}	average amount of required resources for the tasks received from outside the federation	1000	-
u	unit price	100	-

Table 5. Experiment parameters.

	Experiment ID							
Parameters	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
a_r	4	4	4	4	4	4.40	4	4
a_n	6	6	6	6	6	6.60	6	6
d	3	1.10	3	3	3	3	3	3
f	0.40	20	20	20	20	20	20	20
tr	1	1	0/1	0/1	0/1	0/1	1	1
rep	1	1	0/1	0/1	0/1	0/1	1	1
β	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
γ	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
t_{order}	2.5	2.5	2.5	1.18	2.5	2.5	2.5	2.5
sim. time	750	750	750	750	750	750	750	1500

Since there are several stochastic parameters, 50 simulation runs were executed for each parameter set, and the average of the results is visualized in the diagrams. In addition, the confidence intervals on 95% confidence level are also mentioned (except Experiment (7) and (8) where a single simulation run is depicted in the figures). The experiment results are presented mainly using two measures:

- *Average task lateness*, which means the sum of differences between the task due date and the real completion time, for each task which was completed by the federation during the simulation run, divided by the number of mentioned tasks. If a task is finished earlier than the due date, the difference is negative – it decreases the average.
- *Average resource utilization*, which is computed by averaging the resource utilizations for all the resource types a specific agent has, in each model time unit.

Average task lateness indicates the performance of the federation as seen from outside – this value is important for outsider companies, who are sending orders to the federation. Average resource utilization is important for the companies inside the federation: they are trying to maximize the utilization of their resources. Traditional metrics – for example throughput or WIP – are not used here, because these two are depending on the frequency and size of the incoming tasks and are not characteristic as for the performance of the federation. The aim of the federation is not to increase the throughput, but to increase the service level: to complete as many received requests as possible on time. If the federation performs better, it won't be able to complete much more tasks, because the order stream is fixed in the presented model.

3.1. Experiment (1) – agent flexibility

First, in order to investigate the flexibility of the agents (this parameter is equal in all company agents in the simulations), some experiments were performed. As mentioned, flexibility is the parameter that determines the amount of additional resources that a company can use in case of reorganizing its production, given by the percentage of original resource amount. For example, a company with 20% flexibility means it can offer 120% of its original resources ($r_{max,n}$) in the processing interval of a specific task, after being asked to reorganize its production. In Figure 7 one can see the percentage of tasks that were

- insourced (the agent had the specific resource and carried out the task by itself),
- completed after sending out (reorganizing or dividing was not necessary),
- completed after reorganizing,
- completed after dividing, or
- marked as failed.

In these experiments, companies divided the requests into three parts (if there were no offers after reorganizing). In case of divided tasks, when visualizing results, the original task amount was considered: if an agent divided the task into three parts, but only one of them was completed, 1/3 was added to the 'completed after dividing' category. One can see in Figure 7 that the rate of tasks completed after reorganizing increases in line with the company

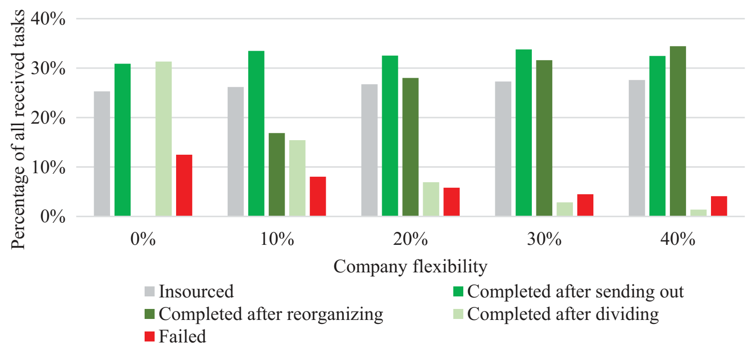

Figure 7. Effect of company flexibility on types of task completion.

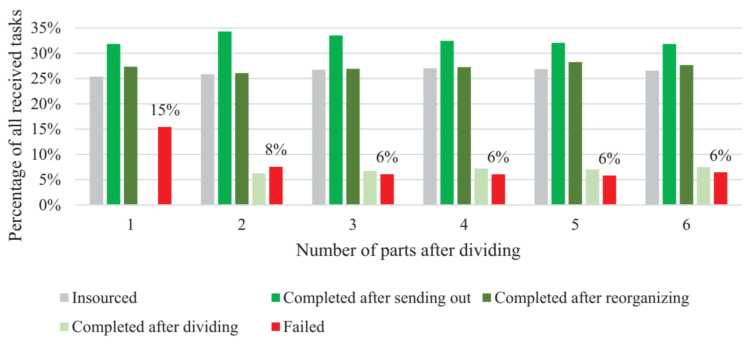

Figure 8. Effect of dividing tasks on types of task completion.

flexibility, and the rate of tasks completed after dividing decreases. The ratio of failed tasks remains almost the same, which means that in the described system, if a company cannot find an offer after asking the other companies to reorganize their production, it will find an appropriate offer after dividing the task. The question may be asked: why does a company try to ask the others to reorganize, if dividing always solves the problem? Why doesn't it divide the task immediately? The answer is, if a company assigns a task to one partner, it can choose the one with the best parameters (unit price, trust, reputation), and does not have to compromise with agents having weaker features. Besides, dividing tasks causes additional costs in reality (for example transportation). In the following series of experiments – as one can see in Table 5 – 20% flexibility was considered at all companies. As mentioned, 50 simulation runs were performed for each parameter set – the confidence intervals on 95% confidence level are between 0.14% and 0.17% of all received tasks in case of each task completion type.

3.2. Experiment (2) – dividing tasks

Some experiments were performed to determine the appropriate number of parts the agents can divide the requests into. As one can see in Figure 8, no dividing leads to a higher rate of failed tasks (16%). In the following experiments, dividing tasks into three parts was set, because after this value the ratio of failed tasks remains the same (6%). Here, the confidence intervals on 95% confidence level are between 0.13% and 0.18% of all received tasks in case of each task completion type.

3.3. Experiment (3) – effect of considering trust/reputation in decision-making

In this experiment, the effect of considering trust and reputation was investigated. Four different cases were simulated: agents could use both trust and reputation values, one of them, or none of them to choose the best offer. According to the results, when making decisions between offers it is worth to take at least one of them into consideration because the average task lateness is lower in these cases (Figure 9). According to the experiments, in the models in which agents take reputation values into account, the federation performs better than in the other cases where only trust values or none of them were considered. There is only a little difference between considering only reputation and considering both trust and reputation (the difference is smaller than the deviation of the results). The reason for this is that reputation more accurately determines the reliability of an agent because it is calculated on the base of a higher number of tasks, as it is formed by interaction with all the other agents. The confidence intervals on 95% confidence level are between 0.03 and 0.04 model time units in case of each model type.

3.4. Experiment (4) – effect of incoming order frequency

In this series of experiments, the effect of the change in the incoming order time period – which can be interpreted as the load of the federation – was investigated. The question was, how the overall system performance changes if the federation members receive orders (with the same processing interval and amount of required resources) more often, therefore the load of the enterprises increases. In Figure 10 the incoming order time period was changed between 1 and 18, and the simulation was run applying the normal and the advanced model, as well. The average task lateness is visualized in Figure 10 in each case for the normal and the advanced model, and the difference between them is also shown. The confidence intervals on 95% confidence level are between 0.8 and 2 percentage of the average task lateness value in each case.

As one can see in Figure 10, as the time period between two incoming orders increases (in other words the load of the federation member decreases), the average task lateness decreases as well in case of the advanced model. This is because in a federation where the agents consider trust and reputation in decision-making, the agents who are working faster and more reputable, win more tasks, thus the average of task lateness is lower. In case of the normal model, when the task arrival time period reaches around five

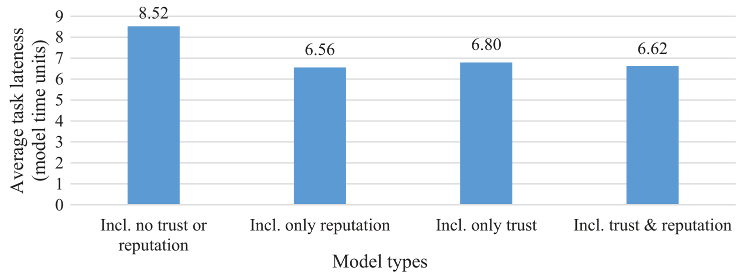

Figure 9. Effect of considering trust and/or reputation on average task lateness.

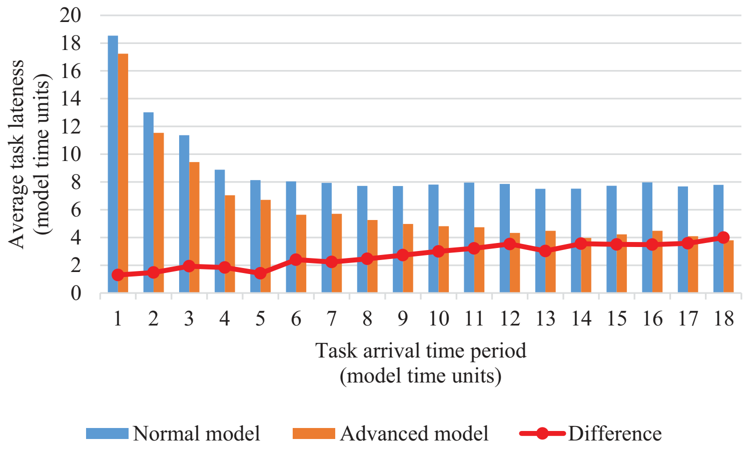

Figure 10. Effect of changing the incoming order time period on average task lateness.

model time units, average task lateness decreases slower than before. This is because here is the point where the resources of the agents become fully utilized, and if the agents receive orders less frequently, they will always have enough resources to finish tasks with lateness max. four model time units. In the normal case, all the agents have the same average utilization rate for their resources, because here the decision-making is made based on the response time that is uniformly distributed.

In case of the advanced model, the level mentioned above is around six model time units: after this, if the load of the federation decreases, the average task lateness decreases much slower than before. If the average resource utilization is investigated in case of the advanced model, this is not the point where the resources of the reliable agents get fully utilized, but the point from which the difference is gradually decreasing between the resource utilization of reliable and non-reliable agents; more and more tasks are performed by the non-reliable agents if the load of the federation is increased.

3.5. Experiment (5) – differences in resource utilization

In this experiment, the two model types are compared based on the average resource utilization of the agents. In Figure 11, the non-reliable C01-C06 companies and the reliable C07-C10 companies are participating in the federation. In case of the normal model, all of them have resources utilized between 40% and 50% (since the decision-making is based on response time), while in the advanced case the higher utilization of reliable companies' resources is clearly visible. The confidence intervals on 95% confidence level are between 1 and 2.5 percentage of the average resource utilization values in each case.

3.6. Experiment (6) – federation size and trustfulness

In Experiment (6) the federation size was increased from 10 to 100 companies, and the average task lateness was compared in case of the normal and the advanced model. Figure 12 shows the results: as the federation grows, the difference between the two models gets larger: trust and reputation have more effect on the average task lateness. The confidence intervals on 95% confidence level are between 1 and 2 percentage of the average task lateness values in each case.

A little fluctuation can be noticed in Figure 12 between federation sizes which are divisible by 10 and the other values. This is because in this experiment when increasing the number of members in the federation, the 10 company agents that are introduced in Table 2 were duplicated – for example in case of 50 agents, 5 agents had the same parameters as C01 in Table 2. In the other cases, when the number of members is not divisible by 10, non-reliable agents similar to C01-C05 were added to the federation, this way increasing the rate of the non-reliable agents, and increasing the average task lateness, too.

3.7. Experiment (7) – change of trustfulness in time

Here the change of reputation is investigated during the simulation run, applying 10 company agents

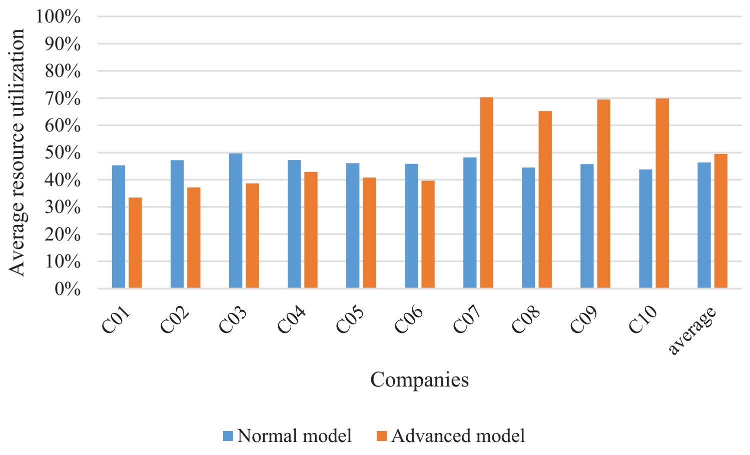

Figure 11. Average resource utilization in the normal and advanced model.

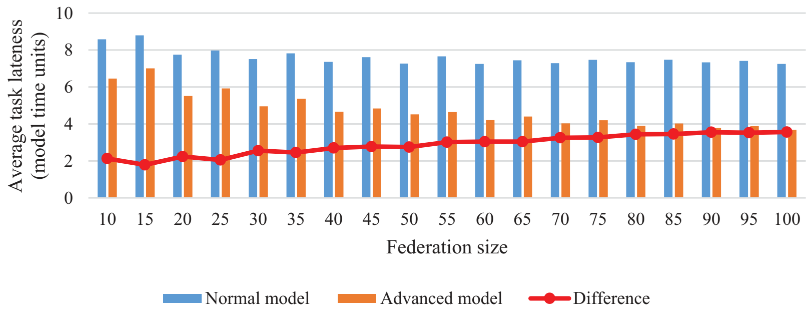

Figure 12. Effect of considering trustfulness in different federation sizes.

included in Table 2. As mentioned, C01-C06 are non-reliable ones, C07-C10 are the companies who are reliable. The difference between them is clearly visible in Figure 13 – the non-reliable agents are marked with bright colours and the reliable ones with darker colours. The reason for the authors chose reputation to compare companies is that reputation is more accurately determines the reliability of an agent since it is calculated on the base of higher number of tasks, as it is formed by interaction with all the other agents.

In Figure 13, at the beginning of the simulation all agents are on the same reputation level (80), until the first tasks are not finished. After then, the two groups are dividing from each other: the reliable agents' reputation values are changing between approximately 75 and 90, the non-reliable ones are between 55 and 70. The values are fluctuating due to the stochasticity of task lateness, but the boundary between them is clearly visible. This experiment has the same parameters with the one visualized in Figure 11: as one can see, the difference between the reputation values has an effect on the resource utilization values, as well.

3.8. Experiment (8) – an unexpected negative event

In this experiment, the effect of a sudden change in trust and reputation values were investigated. In reality, due to some political or economic news or other unexpected event the community's opinion could suddenly change in a negative manner about a certain company. The effect of such an event is simulated by decreasing the reputation value and all the other company's trust value about the reliable C09 to 10, at model time 300. The results can be seen in Figure 14, where the reputation of all agents and the resource utilization of C09 are visualized: it lasts around 100 model time units for C09 to reach approximately the same reputation level as it reached before. This recovery time can be influenced by γ smoothing factor – according to Equation (5), the

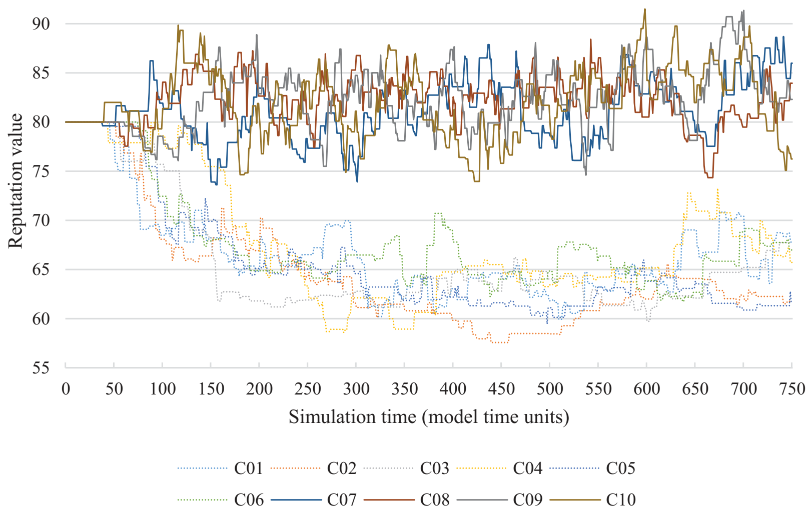

Figure 13. Change of reputation values during the simulation run.

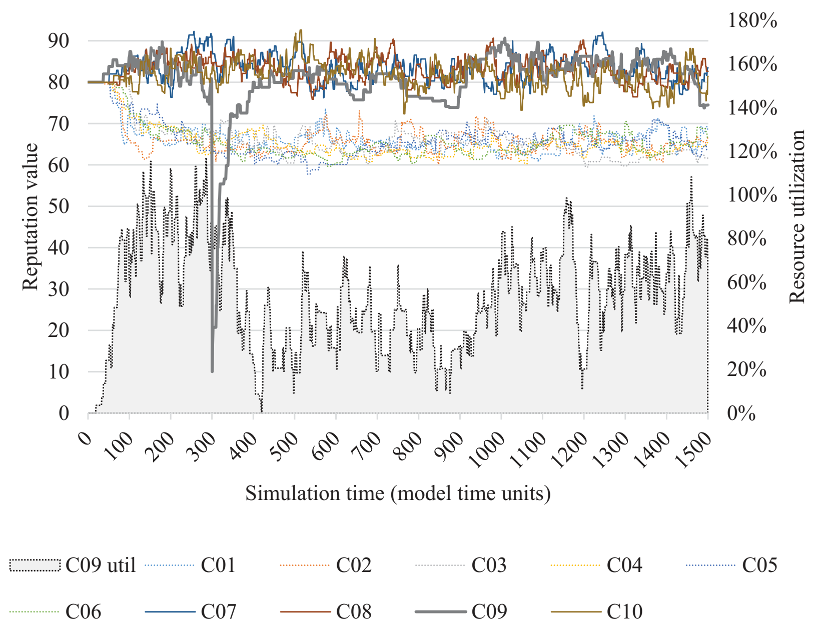

Figure 14. Effect of an unexpected event to reputation values and resource utilization.

higher γ are, the older reputation values count the less. This way, a trade-off between appreciating the positive long-term performance and penalizing the temporary bad performance can be set in the model. As one can see in Figure 14 the resource utilization of C09 is not increasing along with the reputation, because this unexpected negative event also affects the trust values. The recovery time in case of trust values is influenced by β smoothing factor according to Equation (4), the similar way as reputation values were affected.

Figure 15 shows all the other agents' trust values in connection with C09 during the simulation run: as it can be seen subjective trust values are increasing much slower than the public reputation after the event with negative effect. Reputation is influenced by all the finished tasks, thus all the interactions finished by C09 can increase this value – therefore it increases faster according to the agent's performance. In contrast, trust values can increase based on fewer tasks, only if there was an interaction between C09 and the specific agent. This way, when the agents are evaluating offers sent by C09, they count with the high reputation and the much slower increasing trust values. That is why resource utilization of C09 reaches the original level after hundreds of model

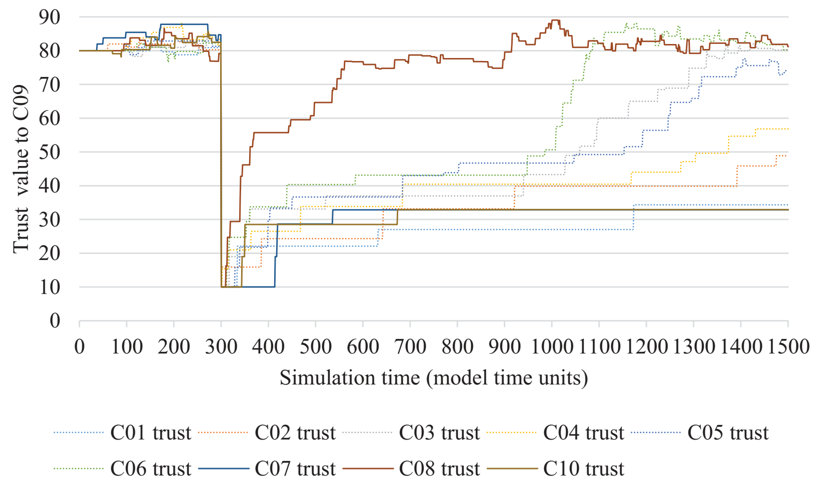

Figure 15. Effect of an unexpected event to trust values to a company agent (C09).

time units after the event with a negative effect. In this case, re-running the simulation naturally leads to different diagrams due to the stochasticity of some model parameters – but the trends were the same as presented in each experiment.

4. Conclusions and future work

In the paper, a distributed manufacturing resource sharing mechanism was introduced, where trust and reputation also were taken into consideration in decision-making when selecting from resource offers – in order to reward agents who are keeping their promises and penalize who are not. Since the main pillar of a distributed manufacturing system is for each participant to have a strong commitment to its promises, it is essential to make a difference between reliable and non-reliable partners. If the participants are not incited to keep their promises, the performance of the federation can decrease. In the presented model, resource offering agents are able to reorganize their production with the aim of fulfiling additional tasks, and resource requesting agents can divide tasks to increase the probability of successful outsourcing as well – this way operating in a more flexible way. Agents choose the best offer based on subjective trust, public reputation values and unit price, where the former two are dynamic features of agents derived from their performance in connection with keeping task due dates, the latter is their static, pre-defined property. There is also a possibility for an agent to cancel a task it promised to complete: this behaviour is penalized by decreasing trust and reputation values. With considering trust and reputation agents can differentiate between reliable partners who are keeping their promises and non-reliable ones who are keeping the due dates in a lower extent and refusing undertaken tasks in a higher extent.

Multi-agent simulation experiments were run to investigate the overall system performance when applying the suggested model. Based on the experiments, if trust and reputation are considered in decision-making (advanced model), the system performs better than in case when offer evaluation is based on a static parameter only (normal model). The difference between the advanced and the normal model (in other words, the impact of considering trustfulness) depends on the federation load: if the participants are highly overloaded and some of the agents are forced to work together with non-reliable partners (because they want to complete the received orders), the difference is smaller than in case of a less loaded federation, where there is the opportunity to choose a more reliable partner to work with. The results have also shown that the higher the number of the federation members is, the higher the impact of trustfulness is on the federation performance. It was also presented that considering the trust and reputation of the participants affects the utilization of their resources as well: reliable agents' resources are utilized on a higher level. The effect of an unexpected negative event has been tested, too: it is easier to build up good (public) reputation than to recover from bad trust values in the applied model; and this causes the low utilization level of resources for a relatively long time after the negative effect, too. Results presented in this paper suggest that including trust and reputation in a manufacturing resource sharing mechanism really makes a difference regarding the performance of a federation containing manufacturing companies and set the ground for further investigations.

In future works, the model will be extended by considering that agents are able to manipulate the system for selfish reasons: e.g. devaluate a partner in spite of its high reliability, with the aim of using the specific partner's resources more frequently (as other agents will not use them because its low trust and reputation values). Another issue is that a group of participants intentionally overrate a malicious agent to make it win more tasks. It is also a challenging research topic that how can agents be incited to be honest, and how can agents protect themselves against malicious members of the federation, if manipulation is allowed. The resource sharing mechanism can also be realized in a more detailed way: orders could be more complex, consisting of interdependent tasks, each requiring different types of resources – in this case, if a certain task is not finished on time, it causes delays in the completion of a manufacturing job.

Nowadays info-communication technologies allow the cooperation between companies, and to record and update values that are necessary to operate a system that helps participants to make decisions on the basis of trustfulness. The results presented in this paper justify the need to continue the research in connection with trust and reputation systems that are applicable in the production and manufacturing area.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the Hungarian National Research, Development and Innovation Office (NKFIH) under Grant GINOP-2.3.2-15-2016-00002 and Grant NKFIA ED_18-22018-0006.

ORCID

Ádám Szaller http://orcid.org/0000-0001-9555-4070

References

Arrais-Castro, A., M. L. R. Varela, G. D. Putnik, R. A. Ribeiro, J. Machado, and L. Ferreira. 2018. "Collaborative Framework for Virtual Organisation Synthesis Based on a Dynamic Multi-criteria Decision Model." *International Journal of Computer Integrated Manufacturing* 31 (9): 857–868. doi:10.1080/0951192X.2018.1447146.

Becker, T., and H. Stern. 2016. "Impact of Resource Sharing in Manufacturing on Logistical Key Figures." *Procedia CIRP* 41: 579–584. doi:10.1016/j.procir.2015.12.037.

Borschev, A. 2013. *The Big Book of Simulation Modeling: Multimethod Modeling with AnyLogic 6*. Chicago, IL: Anylogic North America.

Chang, L., Y. Ouzrout, A. Nongaillard, A. Bouras, and Z. Jiliu. 2014. "Multi-criteria Decision Making Based on Trust and Reputation in Supply Chain." *International Journal of Production Economics* 147 (Part B): 362–372. doi:10.1016/j.ijpe.2013.04.014.

Cheikhrouhou, N., M. Pouly, and G. Madinabeitia. 2013. "Trust Categories and Their Impacts on Information Exchange Processes in Vertical Collaborative Networked Organisations." *International Journal of Computer Integrated Manufacturing* 26 (1–2): 87–100. doi:10.1080/0951192X.2012.681913.

Chen, T.-Y., Y.-M. Chen, H.-C. Chu, and C.-B. Wang. 2008. "Distributed Access Control Architecture and Model for Supporting Collaboration and Concurrency in Dynamic Virtual Enterprises." *International Journal of Computer Integrated Manufacturing* 21 (3): 301–324. doi:10.1080/09511920701196950.

Fictiv Online Manufacturing Platform. Accessed 10 September 2019. https://www.fictiv.com

Freitag, M., T. Becker, and N. A. Duffie. 2015. "Dynamics of Resource Sharing in Production Networks." *CIRP Annals - Manufacturing Technology* 64 (1): 435–438. doi:10.1016/j.cirp.2015.04.124.

Gartner Glossary: Capable-to-Promise (CTP) Systems. Accessed 20 February 2019. https://www.gartner.com/it-glossary/capable-to-promise-ctp-systems

Hohmann, C., and T. Posselt. 2018. "Design Challenges for CPS-based Service Systems in Industrial Production and Logistics." *International Journal of Computer Integrated Manufacturing* 32 (4–5): 329–339. doi:10.1080/0951192X.2018.1552795.

Hou, Y., X. Wang, Y. J. Wu, and H. Peixu. 2018. "How Does the Trust Affect the Topology of Supply Chain Network and Its Resilience? an Agent-based Approach." *Transportation Research Part E: Logistics and Transportation Review* 116: 229–241. doi:10.1016/j.tre.2018.07.001.

Kádár, B., P. Egri, G. Pedone, and T. Chida. 2018. "Smart, Simulation-based Resource Sharing in Federated Production Networks." *CIRP Annals - Manufacturing Technology* 67 (1): 503–506. doi:10.1016/j.cirp.2018.04.046.

Kaihara, T., Y. Katsumura, Y. Suginishi, and B. Kádár. 2017. "Simulation Model Study for Manufacturing Effectiveness Evaluation in Crowdsourced manufacturing." *CIRP Annals - Manufacturing Technology* 66 (1): 445–448. doi:10.1016/j.cirp.2017.04.094.

Kaihara, T., N. Nishino, K. Ueda, M. Tseng, J. Váncza, P. Schönsleben, R. Teti, and T. Takenaka. 2018. "Value Creation in Production: Reconsideration from Interdisciplinary Approaches." *CIRP Annals - Manufacturing Technology* 67 (2): 791–813. doi:10.1016/j.cirp.2018.05.002.

Kumara, S. R. T., Y.-H. Lee, and K. Chatterjee. 2002. "Distributed Multiproject Resource Control: A Market-based Approach." *CIRP Annals - Manufacturing Technology* 51 (1): 367–370. doi:10.1016/S0007-8506(07)61538-8.

Lanza, G., K. Ferdows, S. Kara, D. Mourtzis, G. Schuh, J. Váncza, L. Wang, and H.-P. Wiendahl. 2019. "Global Production Networks: 'Design and Operation'." *CIRP Annals – Manufacturing Technology* 68 (2): 823–841. doi:10.1016/j.cirp.2019.05.008.

Lanza, G., S. Peters, and H.-G. Herrmann. 2012. "Dynamic Optimization of Manufacturing Systems in Automotive Industries." *CIRP Journal of Manufacturing Science and Technology* 5 (4): 235–240. doi:10.1016/j.cirpj.2012.09.002.

Leitao, P. 2009. "Agent-based Distributed Manufacturing Control: 'A State-of-the-art Survey'." *Engineering Applications of Artificial Intelligence* 22 (7): 979–991. doi:10.1016/j.engappai.2008.09.005.

Li, S., Y. Fan, and L. Xitong. 2011. "A Trust-based Approach to Selection of Business Services." *International Journal of Computer Integrated Manufacturing* 24 (8): 769–784. doi:10.1080/0951192X.2011.574235.

Liu, Y., L. Zhang, F. Tao, and L. Wang. 2015. "Resource Service Sharing in Cloud Manufacturing Based on the Gale–Shapley Algorithm: Advantages and Challenge." *International Journal of Computer Integrated Manufacturing* 30 (4–5): 420–432. doi:10.1080/0951192X.2015.1067916.

Matt, T. D., E. Rauch, and P. Dallasega. 2015. "Trends Towards Distributed Manufacturing Systems and Modern Forms for Their Design." *Procedia CIRP* 33: 185–190. doi:10.1016/j.procir.2015.06.034.

Mourtzis, D. 2011. "Internet Based Collaboration in the Manufacturing Supply Chain." *CIRP Journal of*

Manufacturing Science and Technology 4 (3): 296–304. doi:10.1016/j.cirpj.2011.06.005.

Pei, S., J. Zhao, N. Zhang, and M. Guo. 2019. “Methodology on Developing an Assessment Tool for Intralogistics by considering Cyber-physical Production Systems Enabling Technologies.” *International Journal of Computer Integrated Manufacturing* 32 (4–5): 406–412. doi:10.1080/0951192X.2019.1605200.

Pinyol, I., and J. Sabater. 2013. “Computational Trust and Reputation Models for Open Multi-agent Systems: A Review.” *Artificial Intelligence Review* 40: 1–25. doi:10.1007/s10462-011-9277-z.

Plethora CNC machining on demand. Accessed 10 September 2019. https://www.plethora.com

Rossit, D. A., F. Tohmé, and M. Frutos. 2019. “Production Planning and Scheduling in Cyber-Physical Production Systems: A Review.” *International Journal of Computer Integrated Manufacturing* 32 (4–5): 385–395. doi:10.1080/0951192X.2019.1605199.

Sabater, J., and C. Sierra. 2005. “Review on Computational Trust and Reputation Models.” *Artificial Intelligence Review* 24: 33–60. doi:10.1007/s10462-004-0041-5.

Scholz-Reiter, B., F. Wirth, T. Makuschewitz, and M. Schönlein. 2011. “Robust Capacity Allocation in Dynamic Production Networks.” *CIRP Annals - Manufacturing Technology* 60 (1): 445–448. doi:10.1016/j.cirp.2011.03.035.

Shapeways Portal. Accessed 10 September 2019. www.shapeways.com

Shi, S. Y., R. Mo, H.-C. Yang, and Z.-Y. Chang. 2007. “An Implementation of Modelling Resource in a Manufacturing Grid for resource Sharing.” *International Journal of Computer Integrated Manufacturing* 20 (2–3): 169–177. doi:10.1080/09511920601020805.

Swiss Virtuellefabrik. Accessed 10 September 2019. https://www.virtuellefabrik.ch

Ter Huurne, M., A. Ronteltap, R. Corten, and V. Buskens. 2017. “Antecedents of Trust in the Sharing Economy: A Systematic Review.” *Journal of Consumer Behaviour* 16 (6): 485–498. doi:10.1002/cb.1667.

Váncza, J., and A. Márkus. 2000. “An Agent Model for Incentive-based Production Scheduling.” *Computers in Industry* 43: 173–187. doi:10.1016/S0166-3615(00)00066-X.

Váncza, J., L. Monostori, D. Lutters, S. R. Kumara, M. Tseng, P. Valckenaers, and H. Van Brussel. 2011. “Cooperative and Responsive Manufacturing Enterprises.” *CIRP Annals - Manufacturing Technology* 60 (2): 797–820. doi:10.1016/j.cirp.2011.05.009.

Yan, K., Y. Cheng, and F. Tao. 2016. “A Trust Evaluation Model Towards Cloud Manufacturing.” *The International Journal of Advanced Manufacturing Technology* 84: 133–146. doi:10.1007/s00170-015-8002-5.

Yang, X., S. Wang, B. Yang, M. Chi, and L. Kang. 2019. “A Service Satisfaction-based Trust Evaluation Model for Cloud Manufacturing.” *International Journal of Computer Integrated Manufacturing* 32 (6): 533–545. doi:10.1080/0951192X.2019.1575982.

Zhang, L., Y. Luo, F. Tao, B. H. Li, L. Ren, X. Zhang, H. Guo, Y. Cheng, H. Anrui, and Y. Liu. 2014. “Cloud Manufacturing: A New manufacturing Paradigm.” *Enterprise Information Systems* 8 (2): 167–187. doi:10.1080/17517575.2012.683812.