Towards Engineering Manufacturing Systems for Mass Personalisation: A Stigmergic Approach

Rotimi Ogunsakin*a*, Cesar A. Marin*b*, and Nikolay Mehandjiev*c*

*a*Alliance Manchester Business School, Faculty of Humanities, University of Manchester, Booth Street East, Manchester, UK, M13 9SS. Lecturer.

*b*Information Catalyst for Enterprise Ltd, Haslington, Crewe, UK, CW15QR. Senior Computer Scientist.

*c*Alliance Manchester Business School, Faculty of Humanities, University of Manchester, Booth Street East, Manchester, UK, M13 9SS. Professor of Enterprise Information Systems.

Corresponding author: Rotimi Ogunsakin, rotimi.ogunsakin@manchester.ac.uk

Towards Engineering Manufacturing Systems for Mass Personalisation: A Stigmergic Approach

## Abstract

Mass personalisation is characterised by unpredicted changes in product design and manufacturing processes, as a result of customers’ ability to co-create and co-design products based on personal preferences. Previous approaches that address such unpredicted changes rely on heterarchical control structures, and require changes in manufacturing systems’ layout. However, in mass personalisation context, the notion of planning before production is redundant, hence layout reconfiguration is frequent and at such short intervals that present systems are unsuitable. This paper seeks to deliver distinct production mechanisms which react to the frequent requirement for layout changes, inspired by naturally occurring self-organising systems with stigmergic self-organising mechanisms. These systems lack initial structural organisation, and structure only emerges through reinforcement of local interactions with the environment without the need for planning, centralised control, direct communication and simultaneous presence. Therefore, a manufacturing system for mass personalisation with mobile production resources, heterarchical control architecture, and a stigmergic control mechanism is proposed. Simulation is developed using realistic shoe personalisation scenario, which demonstrates the system’s capability to produce personalised shoes and automatically reconfigure the layout of its production resources.

***Keywords*** — Stigmergy, Flexible manufacturing, Mass personalisation, Ant colony optimisation

# Introduction

Shorter product life cycle, constant changes in consumer needs and behaviour, and high marketplace competition are pushing manufacturing companies to innovate constantly in manufacturing designs and production processes. Also, the increased number of consumers seeking personalisation shows that manufacturing companies need to rethink their manufacturing design paradigm by engineering manufacturing systems with mass personalisation capability (Koren, Shpitalni, Gu, & Hu , [2015](#_bookmark46); Delloite, [2019](#_bookmark28)).

Mass personalisation entails the integration of customers into the product design process (Hsiao & Chiu, [2014](#_bookmark43)), and the absence of appropriate techniques to forecast consumers’ preferences implies that the required production planning and scheduling have to be generated and executed in real-time. Also, the inability to predict the timing and specifics of customer demands makes it extremely difficult to have a fixed layout that is guaranteed to minimise the cost of material handling, resulting in an ever-increasing production cost.

A classic mass personalisation scenario is characterised by distinct product designs, stochastic order arrival, and continuously fluctuating production cost resulting from material handling inefficiencies (Hsiao & Chiu, [2014](#_bookmark43)). Distinct product designs can be addressed through the introduction of multifunctional and flexible production machines (Raj, Ravindran, Saravanan, & Prabaharan, [2014](#_bookmark63)). Stochastic order arrival can be addressed through the use of appropriate shopfloor control mechanism (Sallez, Berger, & Trentesaux, [2009](#_bookmark65); Peeters et al., [2001](#_bookmark59); Gao, Luo, & Yang, [2005](#_bookmark36); Onori, Lohse, Barata, & Hanisch, [2012](#_bookmark57)). However, routing different products for distinct operations on the shopfloor, and managing increasing production costs resulting from the sub-optimal layout of facilities, require that the system’s layout adapts in real-time to the resulting changes in product mix.

Previous approaches that address unpredictable changes of production plans and shopfloor schedule are underpinned by heterarchical control structures, and planning and process flexibilities (Barbosa, Leita˜o, Adam, & Trentesaux, [2015](#_bookmark20); Pujo, Broissin, & Ounnar, [2009](#_bookmark61); Pujo, Dubromelle, & Ounnar, [2010](#_bookmark62)). However, these approaches are based on a conveyor system and static production resources, which limits the flexibility that could be achieved if the layout could be changed dynamically, making them unfit for mass personalisation.

Taking inspiration from current research in industry, such as from Bosch ( [2016](#_bookmark25)) Rexroth ([2018](#_bookmark64)) and Kuka ( [2019](#_bookmark0)), and from the collective behaviour of naturally occurring self-organising systems, this paper proposes a manufacturing system for mass personalisation called Self-organising Manufacturing System for Mass Personalisation (SMS4MP), where the shopfloor layout dynamically emerges to accommodate personalised orders. The system is connected and synchronised with the shopfloor entities, driving them to adapt to production needs in real-time and accommodating mass personalisation.

In SMS4MP, static production machines are replaced with mobile and autonomous production machines capable of changing their layout in real-time. Material handling systems, like conveyor systems, are replaced with mobile and intelligent robots with an integrated pallet for material handling. Simulation is used to evaluate our approach under realistic mass personalisation scenario. The simulation is developed following the recommendation for designing and simulating stigmergic based production control proposed by Van Brussel, Wyns, Valckenaers, Bongaerts, and Peeters ([1998](#_bookmark73)), and Kruger and Basson ([2019](#_bookmark47)). The details of the implementation and simulation results are presented, including implication for industry and research in stigmergy-based control in engineering manufacturing system for mass personalisation.

# Self-organising Manufacturing System For Mass Personalisation

### Self-organisation, emergence and coordination

Self-organisation is the spontaneous creation of globally coherent patterns at a macro-level out of local interactions at a micro-level (De Wolf & Holvoet, [2005](#_bookmark0); Heylighen, [1989](#_bookmark40), [2001](#_bookmark41)). In computational systems, self-organising systems are modelled as a collection of interacting agents representing diverse components with non-linear interactions, and to an important degree, the overall system evolution is unpredictable and uncontrollable (Di Marzo Serugendo et al., [2004](#_bookmark30); Omicini & Viroli, [2011](#_bookmark56)). This has been the major drawback since predictable outputs are desirable in manufacturing systems (Di Marzo Serugendo, Gleizes, & Karageorgos, [2006](#_bookmark32)). However, the self-organising property where local micro-level interactions eventually lead to global coordination and synergy is attractive to processes where exact solutions are not possible in polynomial time, and approximate solutions are desirable.

Self-organising systems exhibit emergent properties when the coherent macro-level global pattern cannot be reduced to micro-level components (De Wolf & Holvoet, [2005](#_bookmark0); Heylighen, [1989](#_bookmark40)). This implies that the emergent patterns are novel with respect to the individual parts constituting the system. Emergent systems are usually not suitable where specific desired goals are expected; hence the proposal by Di Marzo Serugendo et al. ([2006](#_bookmark32)), that when designing artificial systems, it is necessary to have operational definition and tools to enable such system to produce the desired emergent phenomenon. Such operational role can be played by the environment where the systems’ components interact (Weyns, Schumacher, Ricci, Viroli, & Holvoet, [2005](#_bookmark76)).

However, interactions between the different entities in self-organising systems require coordination, that is, management of the dependencies amongst different activities and entities. These interactions are non-linear and unpredictable (Viroli, Casadei, & Omicini, [2009](#_bookmark74); Heylighen, [2001](#_bookmark41)). Thus, the inherent coordination causing the observable appearance of order at the macro-level is not usually expressive enough to capture all the complex properties. Therefore, to implement a computational system with self-organising and emergent properties, a simple probabilistic mechanisms, called self-organising coordination models are needed to fully capture the coordination dynamics (Omicini, [2013](#_bookmark55)). These should be structured to foster only local interactions; the environment evolves independently; it is time-dependent and probabilistic; and with no centralised control (Omicini & Viroli, [2011](#_bookmark56))

### Stigmergy

The concept of *stigmergy* was proposed by Grasse¨ ([1959](#_bookmark37)) to describe the coordination mechanism used by insects, where work executed by an agent leaves a trace in the environment, and that trace stimulates the performance of subsequent work by the same or other agents in the environment. This behaviour ensures a task is performed in the right order, without planning, control, or direct interaction between agents.

Classic examples of stigmergy are found in (a) the pheromone trail left by ants during food foraging, which stimulates other ants to follow the same path (Dorigo, Bonabeau, & Theraulaz, [2000](#_bookmark35)); (b) the actions of young honeybees in a temperature gradient field, where the action of one honeybee determines the action of another (Schmickl & Hamann, [2010](#_bookmark67); Bodi, Thenius, Szopek, Schmickl, & Crailsheim, [2012](#_bookmark22)); (c) Wikipedia, where every reader is stimulated to improve and expand on the writings of previous contributors (Heylighen, [2016](#_bookmark42)).

Stigmergy provides effective ways of enacting coordination in a self-organising system, where agents do not need to plan, remember their previous actions, communicate, be aware of other agents, be simultaneously present, have an imposed sequence of action, commit to a particular action, or be centrally controlled or supervised (Heylighen, [2016](#_bookmark42)). These characteristics make stigmergy a coordination mechanism of choice for managing and coordinating interactions in computational systems situated in unpredictable environments, such as the SMS4MP.

* + 1. *Stigmergy in ants*

In ant colonies, pheromones are used as an environment-mediated coordination mechanism (Dorigo et al., [2000](#_bookmark35)). During foraging, ants deposit pheromone trail on their way back to the nest from a food source. These pheromones are sensed by other ants, which reinforce a route by following it to the food source. The pheromones evaporate, such that non-reinforced or less frequently reinforced pheromones tend to disappear from the environment faster, leaving the frequently reinforced pheromone paths in the environment. Following this simple rule allows ants to effectively coordinate foraging.

#### Stigmergy in young honeybees

Contrary to the notion of the requirement of a marker or physical environment in stigmergy, stigmergy in the most general sense requires neither mark nor quantities (Heylighen, [2016](#_bookmark42)). An example is the young honeybee’s collective behaviour of aggregation in a temperature gradient field (Schmickl & Hamann, [2010](#_bookmark67); Bodi et al., [2012](#_bookmark22)). A typical natural honeybees’ hive is characterised by complex patterns of temperature fields, where the central brood-nest areas are kept at a comparably higher temperature (32*◦ C −* 38*◦ C*) compared to the honeycomb areas and the entrance area, which have a significantly lower temperature. The temperature in the brood-nest is most favourable for the development of larvae and freshly emerged honeybees.

Experimental observation shows that the young honeybees (1 day old or younger) tend to aggregate in the warmer regions most favourable for their development by following simple rules: (a) the young honeybees move randomly when not in a cluster; (b) are likely to stop when they encounter another honeybee; (c) stay longer in warmer areas (Schmickl & Hamann, [2010](#_bookmark67)). Following these rules and without any form of intelligence, plan, or communication, young honeybees are able to aggregate into the region where the temperature is optimal for their development.

#### Application of stigmergy in manufacturing control

Based on the behaviour of ants in the colony, the ant colony optimisation (ACO) algorithm was developed by Dorigo and Di Caro ([1999](#_bookmark34)). The ACO has been applied in solving different optimisation problems, such as the Travelling Salesman Problem (TSP) (Eyckelhof & Snoek, [2002](#_bookmark0)), optimisation of packet routing in communication networks (Di Caro & Dorigo, [1998](#_bookmark29)), coordination of autonomous robots (Mondada et al., [2004](#_bookmark50)), and coordination of manufacturing systems (Komarudin & Wong, [2010](#_bookmark45); Hani, Amodeo, Yalaoui, & Chen, [2007](#_bookmark38)), including job shop scheduling (Nouiri et al.,[2018](#_bookmark52)), and in Self-reconfigurable Manufacturing System for mass personalisation (Ogunsakin, Marín, & Mehandjiev, [2018](#_bookmark53)).

Pheromone-based shopfloor control was proposed by Peeters et al. ([2001](#_bookmark59)), stigmergy has also been proposed for dynamic routing of active products in flexible manufacturing system by Sallez et al. ([2009](#_bookmark65)). The stigmergic behaviour of young honeybees in temperature gradient field has been applied to model a manufacturing system for mass personalisation (Ogunsakin, Mehandjiev, & Marín, [2018](#_bookmark54)).

# Architecture and Control of SMS4MP

The introduction of multifunctional and flexible production machines and appropriate shopfloor control mechanism have been explored to partially resolve the challenges associated with distinct product design and stochastic order arrival (Van Brussel et al., [1998](#_bookmark72); Bongaerts et al., [2000](#_bookmark24); Gao et al., [2005](#_bookmark36)). However, the major challenge remains how to route different products for distinct operations on the shopfloor, and manage frequent layout reconfigurations requirements resulting from product mix changes. Manufacturers are currently exploring the use of modular and mobile production machines to overcome these challenges, for example -- *“Bosch’s factory of the future”* (Bosch, [2016](#_bookmark26); Rexroth, [2018](#_bookmark64)). Indeed, Kuka is already manufacturing industrial robots with mobility and autonomous capabilities (Kuka, [2019](#_bookmark0)).

The use of modular and mobile production machines is further explored to deliver dynamic layout capability responding to the product evolution in real-time (Hu, [2013](#_bookmark44); Koren et al., [2015](#_bookmark46); Wang, Ma, Yang, & Wang, [2017](#_bookmark75); Bosch, [2016](#_bookmark25); Rexroth, [2018](#_bookmark64)). Present production systems with fixed production machines and rigid conveyor system are only able to handle disruptions that do not require structural or spatial modification, such as rearrangement of production resources. To overcome this, an architectural approach where the assumption of static production machines is relaxed and mobile production machines are introduced is proposed. All conveyor systems and other material handling systems are replaced with mobile and intelligent robots with integrated pallets for material handling. Finally, a stigmergy-based manufacturing coordination and control mechanism is applied.

Application of stigmergy-based concepts in manufacturing control, such as the ant colony optimisation (Dorigo et al., [2000](#_bookmark35); [2004](#_bookmark0); Solimanpur, Vrat, & Shankar, [2004](#_bookmark69)) and “beeclust” algorithm (Schmickl & Hamann, [2010](#_bookmark67); Bodi et al., [2012](#_bookmark22)), requires a significant integration effort. Manufacturing entities such as products and machines cannot be modelled directly as ants. For example, it would be impractical to perform an ant exploration through physically moving machines and products around the factory in order to determine optimal layout for the machines or optimal route for the products undergoing production.

The use of delegate multi-agent system (D-MAS) is proposed to address these challenges (Verstraete et al., [2008](#_bookmark73)), where agents live in a virtual system that is connected and synchronised with a real system in real-time. In the SMS4MP, the manufacturing system is based on 3-layered functional architecture, comprising a physical layer, an information layer and a virtual layer, as shown in Fig. [1](#_bookmark1).

[Figure 1 near here]

* The *physical layer* consists of the physical manufacturing entities represented as holons, referred to as the material sub-holons (M\_holons) (Botti & Adriana Giret, [1997](#_bookmark27); Pujo et al., [2009](#_bookmark61)). The M\_holons consists of the material product (the physical product) called M\_product; the mechanical part of the resources called M\_resource, and the material order called M\_order. The physical layer evolves in material time. This implies that the physical layer is subjected to the natural phenomena and constraints of space and time.
* The *information layer* consists of data about the physical entities. It comprises information sub-holons (I\_holons). The I\_holons consists of I\_product, which contains data about the manufacturing process of the physical product (M\_product), its state model and all traceability information; The I\_resource is connected to and controls the M\_resource; and the I\_order represents a task in the production system. The information layer evolves in real-time with respect to the physical system. The real-time evolution is such that processes, schedules and decisions are made in the information layer before being passed to the physical system for execution. Hence, the information system, which is connected to the physical system, contains data about the function and state of the physical system to which it is connected.
* The *virtual layer* evolves in accelerated time, meaning it evolves in a ”fast-forward” mode by extrapolating from present conditions in the information layer to guide the adaptation of the system’s layout to present and future changes.

The interaction between the three layers is such that the holons in the physical and information layer delegate layout optimisation tasks to the software agents in the virtual layer, seeking an optimal layout of the physical resources in the manufacturing system based on the prevalent product mix on the shopfloor. The outcome is communicated back to the information layer and passed onto the physical entities in the physical layer for execution. Managing the relationship between the physical and the information layer, and between the information layer and the virtual layer requires some form of control mechanisms. The Product, Resource, Order, Simulation for Isoarchy Structure (PROSIS) *holonic control architecture (HCA)* is used for the former (Pujo et al., [2009](#_bookmark61)), while the *delegated multi-agent system (D-MAS)* is used for the latter (Verstraete et al., [2008](#_bookmark73)).

A top-down research approach is followed in analysing the problem and developing a solution (Hevner et al., 2004), while ensuring that the operational principles and dependencies in the context of manufacturing and mass personalisation are considered. Fig. 2 below shows an overview of the top-down approach used.

[Figure 2 near here]

### Physical vs information layers of SMS4MP

The PROSIS HCA (Pujo et al., [2009](#_bookmark61)) is used as a framework for managing the development and operation of the entities in the physical and information layers referred to as holons, which constitute the physical and information part of the manufacturing system (*see* Fig [1](#_bookmark1)). The PROSIS holonic architecture provides a set of rules, guidelines, and specifications for managing the development, operation and interaction of the holons in the manufacturing system.

The robots and the on-board products, the production machines, and the load and unload stations are represented as the product, resource and order holons respectively. Each of these holons is autonomous, have decision capacity, communicate, and cooperate. Holons in the manufacturing system are treated as a conceptual entity that is based on the association of a given material structure called material holon (M\_holon) that functions in the physical layer; an informational and processing system called information holon (I\_holon) that provides connection, decisional intelligence, and functions at the information layer (Pujo et al., [2009](#_bookmark61)) (See Fig. [3](#_bookmark2)).

The isoarchic characteristic of PROSIS brings about self-organisation in the holonic system. This implies controls decision are made locally and therefore, communication, cooperation and control mechanism is required. The Autonomous Control Entity (ACE) is proposed for this purpose, where it supports the interaction and emergence of solutions to disturbances in the production system. Further review on ACE, and abstraction of manufacturing resources as holons in the context of PROSIS can be found in Pujo et al. ([2009](#_bookmark61)).

[Figure 3 near here]

### Information vs Virtual layer of SMS4MP

Each holon in the physical layer is projected and delegated as a software agent in the virtual layer. These are referred to as product-agents, resource-agents, and order-agents. The stigmergic behaviour of the product-agents and resource-agents in the virtual layer is underpinned by the stigmergic behaviour of ants and young honeybees in colonies.

Hence, the product-agents are represented as product-ants, whereas two different types of resources are implemented in the virtual layer using two different instances. In one, the resource-agents are represented as resource-ants, and in the other, they are represented as resource-bees. Each of the holons in the physical layer delegates a task to these software agents in the virtual layer. In the context of SMS4MP, an exploration task is delegated to find an optimal location for the mobile resources based on the prevalent product mix on the shopfloor.

The software agents in the virtual layer evolve in accelerated time; therefore, possess future information on the product to be produced. The different holons in the information layer (*I\_*holons) possess data about the present state of the physical entities in the physical layer (*M\_*holons). In addition, they possess information about orders already placed but yet to be allocated to a product holons. The information about future order is combined with the information about the state of the physical resource to create and delegate the software agents to find an optimal location for the resources at a future time. The result of the optimisation task is sent to the physical layer through the information layer for execution.

### Delegate-MAS and stigmergic Approach in SMS4MP

The delegate multi-agent system (D-MAS) (Verstraete et al., [2008](#_bookmark73)) is used to coordinate the interaction between the information entities (I*\_*holons) in the information layer and the software agents in the virtual layer. The D-MAS consists of three elements: the *holons*, the *software* *agents*, and the *simulation* *environment*.

The *Simulation environment* is a software representation of the manufacturing environment. It constitutes the virtual layer and are modelled as a complete graph, where the nodes represent the location of the resources and the edges represent the connection between the resources. Given that these resources are mobile, the edges can be said to be flexible. Therefore, the nodes and edges are constantly evolving. The simulation environment also consists of pheromone infrastructure in the form of a shared data-space, where ant-agents deposit, observe, and modify pheromones in the simulation environment.

The I*\_*products create and delegate tasks to the product-agents in the simulation environment, while the I*\_*resources create and delegate tasks to resource-agents in the virtual environment. The I*\_*resources and I*\_*products are responsible for maintaining the population and diversity of the software agents in the simulation environment and interpreting the results of reported delegated task. The I*\_*product is capable of delegating a single task to multiple product-agents. In our implementation, the I*\_*product delegates one task to multiple product-agents in the virtual environment, where each explores the problem space for the best solution using digital pheromones. On the other hand, the I\_resource delegates one task to one resource-agent only.

*Software agents* are represented as product-ants and resource-ants or resource-bees. They execute delegated tasks by cooperatively exploring possible solutions in the simulation environment. Through the use of an ant-based optimisation algorithm, approximate solutions are found and reported back to the issuing holon. The different software agents represent a software-model of the issuing holon, such that after creation and delegation of task, the resulting software agents are autonomous and are not controlled by the issuing agents, but conform with the same behavioural constraints as that of the issuing agent. In addition, when a resource-bee is used instead of a resource-ant in a separate instance, this implies that the two delegated software agents in such instance are product-ants and resource-bees. These two agents cooperate, using distinct but complimentary stigmergic mechanism to achieved their optimisation goal.

*Relationship between the different elements* is such that the holons, which are the I\_products and I\_resources, create and delegate software agents at regular intervals based on the order arrival rate and product mix in the physical layer. The I\_product delegates task to multiple product-ants to enable cooperation during the exploration of the environment as the ant-based heuristic algorithm used in the simulation environment is population base. Hence, to arrive at the optimal solution, the best solutions are promoted through pheromone reinforcement. Whereas, an I\_resource delegates task to one resource-ant or resource-bee, as the optimisation task is performed by the product-ants, while the resource-ants or resource-bees augment the optimisation process for speedy convergence as will be explained later.

#### Stigmergic coordination and control mechanism in SMS4MP

In the virtual layer, the product-ants and resource-ants follow two separate but complementary stigmergic behaviours. The product-ants hold the process plan of the product they are delegated to produce, search for the appropriate resource-ants to execute operations in their process plan, and share knowledge of the possible location of resource-ants with other product-ants using pheromones. The resource-ants, on the other hand, move to a location called *hotspot*, a location that increases the probability of being discovered by the appropriate product-ant. The goal of the resource-bee is also the same as that of the resource-ants but achieved using different stigmergic strategy.

The stigmergic techniques used by the different software agents in the virtual layer to arrive at the optimal location of the resources are summarised below.

#### Product-ants

* + - 1. A *Product-ant* discovers resources for required operations either by a random search or by indirect interaction with other *product-ants* through pheromones.
			2. There are no pre-defined construction-graphs which show the list of available possible paths from *product-ants* to *resource-ants* in the simulation environment. Therefore, *product-ants* randomly search the simulation environment for *resource-ants* and construct a graph where the nodes represent the resources used, and the edges represent the distances between these resources. The constructed graph is used to compute a *cost C* equivalent to the material handling cost for producing one product.
			3. *Product-ants* only make use of private and local information available within their cognitive-range, which is a range within the *search-space* for which the *product-ant* is capable of sensing *pheromones* deposited by other *product-ants* in the simulation environment.
			4. Pheromone variables are created when a *product-ant* discovers the desired resource by a random search, or are reinforced when such resource is discovered through the use of pheromone trail deposited by other product-ants.
			5. Pheromone variables are assigned a *3-tuple* Θ *→* (*lp*, *t*, *β*), where *lp* is the last location of the *resource-ant* used by the *product-ant*; *t* is the time when the *resource-ant* was used; and *β* is the pheromone type equivalent to the type (operations type) of the *resource-ant*.
			6. *Product-ant* has a speed *s*, which is the distances advanced per simulation step, and a memory *Mθ* for storing all its accumulated *experiences*, which are the graphs representing the different *resource-ants* used for the required operations.
			7. Pheromones evaporate with a constant evaporation rate (*decay-rate*) per simulation step, which is set to a value between 0 and 1. This serves as an exploration mechanism to prevent the system from converging at sub-optimal solutions by making sure older information are replaced with resent information.
			8. When a *product-ant* with speed *s* senses a pheromone at time *t*0:
				1. It determines if it leads to the required *resource-ant* by using the pheromone property *β*.
				2. If it leads to the required *resource-ant*, the *product-ant* computes the shortest path to the *resource-ant* using the location *lp* and follows it. However, meeting the resource at the computed location is probabilistic due to the mobility of the *resource-ant*.
			9. After all production tasks are completed, the *product-ant* computes and deposits the total *cost* of producing the delegated product in a shared data-space, including the details of the constructed graph (resource layout).
			10. *Product-ant* dies afterwards by simply being deleted from the simulation environment.

#### Resource-ants

1. *Resource-ants* are special types of *ants-agents*, such that *pheromones* created by each *resource-ants* are private to the *resource-ant* that creates it and cannot be sensed and updated by other *resource-ants*.
2. *Resource-ants* randomly explore the environment for possible *product-ants* for which they can execute operations. Pheromones are created and deposited at the location (*hotspots*) where such operations are carried out.
3. Pheromones in the context of *resource-ants* are singletons containing the coordinate *lr* of the position where operations are executed previously. These pheromones are stored in a *pheromone-table* private to the *resource-ant*.
4. Resource-ants visit previous locations where operations are executed before resulting into exploration strategy, as the probability of finding *product-ants* at such location is higher than a random exploration of the simulation environment. This is due to the availability of *pheromone-trail* in the environment leading to the location.

#### Resource-bees

1. *Resource-bees* randomly explore the environment for possible *product-ants* for which they can execute operations.
2. *Resource-bees* use the presence of executable operations to determine how long to wait at a location before randomly exploring the environment.
3. After each successful operations, the *resource-bees* wait at the same location for a period *tw*, after which if no new operations are executed, it randomly explores the environment.
4. The probability of a *resource-bee* executing additional operation(s) at the waiting location is higher than other locations due to the presence of pheromones in the environment leading to the same location.

### Layout optimisation strategy

Each delegated *product-ant* constructs a graph where the nodes represent the different resources (*resource-ants*) used for executing different operations, and the edges are equivalent to the distances between these resources. The *product-ants* seek to minimise the sum of the product of flow between the nodes (production machines’ location) and the corresponding distances between the different machines.

The process of evaluating the cost *C* of executing delegated tasks, and selecting the optimal layout corresponding to the minimum cost *C* = *Cmin* that minimises the total material handling cost for the system is as follows:

1. The product-ants explore the system and deposit the cost of executing delegated tasks in a vector , where the lowest cost is selected at a pre-defined time as the final solution, and the corresponding layout becomes the optimal layout until a new minimum cost is obtained at time that is lower than cost at time
2. The layout *V* = *(v*1, *v*2, *· · ·* , *vn−*1, *vn)* : *vi ∈ V* is a vector containing the nodes (locations) of the resources (resource-ants) used by the product-ant.
3. The *product-ants* and the *resource-ants* are assumed to operate in Euclidean space. Therefore, there exists a function *f* that computes the distance between two nodes, where *wi ∈ W* is the weight of the edge from the node *vi* to *vi*+1, which is equivalent to the Euclidean distance between the two nodes or the two resources located at these nodes.
4. The material flow cost *u* between two nodes and is the total number of expected trips between and or between the resources located at *vi* and *vi*+1. The material flow cost *u* is represented in an interdepartmental flow matrix (IDFM) Ogunsakin (2020)
5. The speed *i* = (0, 1) at which the *product-ants* travel from to using the edge is proportional to the material flow cost *u* between the two nodes. This allows the *product-ants* executing production processes, and whose process plans include resources with higher flow cost *u* to explore and respond faster to the delegated task, thereby increasing the possibility of contributing to the final solution.
6. The cost of executing delegated task by a product-ant is defined as follows:



1. The goal of the heuristic approach is to obtain a minimum value of = whose corresponding layout minimises the total material handling cost. However, Eq. [1](#_bookmark3) does not guarantee that the layout *V* corresponding to a minimum cost will be the optimal layout for the system. This is because there is a tendency that the minimum cost is deposited by a *product-ant* whose routing sequence consists of resources with lower material flow cost *u*. This implies that the layout will be optimal for the production of products whose routing sequence corresponds to that of the product, and sub-optimal for the production of products whose routing sequence requires resources with higher material flow cost *u*, which accounts for the greater percentage of the total material handling cost in the system. Therefore, a heuristic technique that ensures the selected optimal layouts are contributed mainly by *products-ants* whose routing sequence consists of resources with higher flow cost is given as:



Eq. [2](#_bookmark4) ensures that the cost *C* of executing a delegated task is either reduced or increased relative to the value of *u* and *s* (given that *s* is proportional to *u)*. *This ensures a higher possibility of selecting an optimal layout contributed by a product-ant whose routing sequence consists of resources with higher material flow cost u, hence, resulting in a layout that is optimal for the whole system*.

1. When the demands for a particular design variation increases, this results to changes in product flow between resources. The system gets trapped in a local minimum, where the subsequent minimum cost obtained at time is persistently greater than the minimum cost , and the layout remained unchanged. This state is detected, and the system reacts using a technique inspired by the guided local search (GLS) algorithm (Voudouris & Tsang, [2003](#_bookmark0)), which is implemented as follows:
	1. If the new minimum cost belongs to a candidate solution , such that *iff* . Where is the threshold that determines a candidate solution for the augmented cost and regulates the sensitivity of the system to the changes in the simulation environment.
	2. If , the optimal cost is redefined as , where is the augmented cost and is the parameter determining how fast the system escapes from local minimum. The augmentation process continues until , where the new layout corresponding to the new minimum cost becomes the new optimal layout for the system.
2. The values assigned to the parameters and are dependent on the goal of the system designer. A lower value of results in high sensitivity leading to frequent layout changes. Whereas, a lower value of results in low sensitivity and less frequent layout changes.

# Simulation Design

The mass personalisation scenario used in this paper is focused on ”design personalisation” (Boe¨r, Dulio, & Jovane, [2004](#_bookmark23)). A personalisation scenario where selected features of a product relating to colour, material combination, decoration and small details can be configured by the customers based on personal preference. This personalisation process is applicable to products whose physical appearances can be modified without changing the structure or the function of the products, for example, automobile (painting), shoe (design), and complimentary cardholder (colours).

Design personalisation for shoe production is formulated by taking inspiration from Di Roma ([2017](#_bookmark33)) and Heredia, Ceballos, & Sanchez-Torres ([2018](#_bookmark39)). Generally, in a typical manufacturing system, a group of resources are required to manufacture different parts, assemble, inspect, sort and package for shipping. However, the scenario used in this research is focused on personalisation process; therefore, six different resources are proposed to execute the different personalisation operations.

The following assumptions are made on SMS4MP functionalities: (a) each resource is multifunctional, meaning it is capable of carrying out different operations; (b) no two resources can execute the same operation; (c) each operation is assumed to be atomic; (d) different operations sequence produces different and unique designs; (e) no two operations sequences are the same; hence, no two product units are the same.

Following from the above, each shoe personalisation task is achieved by appropriately routing products to different resources for different operations. However, two products may have equivalent routing sequence but not necessarily result in the same design due to the multifunctional capability of the resources and the fact that personalisation task is defined by operations sequence and not routing sequence. Therefore, products with similar routing sequence are classified as *product group*, and products with a particular operation sequence are classified as *product unit*.

Simulation models and algorithms are developed and simulated using NetLogo (NetLogo, [2019](#_bookmark0)) (*see* Appendix C). NetLogo has been used to implement complex concepts in manufacturing systems, such as dynamic routing using simulated annealing (Sallez et al., [2009](#_bookmark65)), dynamic architecture for the optimised and reactive control of flexible manufacturing scheduling (Pach, Berger, Bonte, & Trentesaux, [2014](#_bookmark58)), and simulation of multi-agent manufacturing systems (Barbosa & Leitao, [2011](#_bookmark21)).

Two-layered simulation is developed. The first layer represents the real manufacturing system existing in the *real world*, where the actual production takes place, and production entities are represented as holons. This is equivalent to the physical and information layer of the functional architecture (see Fig [1](#_bookmark1)). The second simulation layer represents the virtual system existing in the virtual world and populated by software agents. The two layers (the physical and the real world) are connected for constant exchange of data about state, behaviour, and solutions.

### Formal model of SMS4MP state machine

A low-level architecture of the system is presented in Fig. [4](#_bookmark5) showing the manufacturing system existing in the real world and the virtual system existing in the virtual world. After which a discrete event system specification (DEVS) (Song & Kim, [1994](#_bookmark70); Zacharewicz, [2008](#_bookmark78)) formalism of the system is presented.

[Figure 4 near here]

The different entities have *output* and *input* ports, where the *output* port of one is connected to the *input* port of the other. The dynamics of the system is defined by two state transitions, which are *internal transition* and *external transition*. An *internal transition* occurs when an entity spontaneously changes its internal state after completing an activity and generates an output event. Whereas *external transition* occurs when an entity changes its internal state as a result of an input event through the input port. *External transition* requires that a message, which is generated through an output event of one entity, is converted to an input event of another entity. These messages are treated as variables, instructions or entities like physical products, and are represented as “*∗* info” as shown in Fig. [4](#_bookmark5) ( “\*” represents the input port). Table [1](#_bookmark6) shows the contents of these messages.

[Table 1 near here]

#### Formal model of order-agent’s state machine in SMS4MP

The DEVS model of order-agent’s state machine is presented in Fig. [5](#_bookmark7). It mainly consists of nodes that represent an activity or a state. Dotted arcs represent an internal transition, and solid arcs represent external transition. For completeness, each of the nodes is named to specify the entity carrying out the activity – the thick continuous circle represents the holon (the physical component), the dotted circle represents an external agent, and the continuous circles represents the agent under consideration. Input and output events are represented by “?” and “!” respectively. Where implies that the message is received at the input port , and implies the message is sent through the output port . Port numbers are usually omitted for an internal transition. Variables are represented with the sign as the prefix, while material entities like parts and products are represented with as the prefix. The symbol is used to test for equality, and is used for “difference” test.

[Figure 5 near here]

In Fig. [5](#_bookmark7), the order-agent is created and updated by the order holon and waits in the virtual layer for request from exploring product-ants. When such request is received, the order-agent transitions from “waiting for product request” to “processing product request” by producing the output event “*po\_in f o* received !”. After successful processing, the order-agent transitions to “send product information” and produces an input event (external transition) by sending the result of processing activity to the product-ant. The external transition simultaneously leads to an internal transition “store request transaction” and then finally to “waiting for product request”. The update process between the order holon and the order-agent is continuous; this ensures that the order agent is in sync with the order holon.

#### Formal model of product-ant’s state machine in SMS4MP

In Fig. [6](#_bookmark8), the product holon creates product-ants and delegates task. The product-ants represent a *copy* of the product holon in the virtual layer. Each product-ant sends a product request to the order-agent based on the content of the delegated task and waits for a response. When the product to be personalised is on-board the product-ant, it searches its memory for appropriate production plan for the personalisation task and searches for resources (resource-ants) to execute different operations based on the retrieved plan. The product-ant joins the resource-ant’s queue when found, send operation request and wait for the end of operation before deciding on the next move. After each operation, the product-ant stores each resource-ant’s location and elapse time between successive operations. If all the operations in the product-ant’s plan are executed, all accumulated cost and resource locations are deposited before being deleted from the simulation environment.

The relationship between the product holon and the product-ant is the creation and delegation relation. Therefore, the product holons create a specific number of product-ants and delegate a similar task. The result with the least cost is selected as the optimal solution, and the corresponding resource-ants’ location is selected and executed by the resource holons in the physical layer. The rate at which layout selection is made can be adjusted using the number of distinct delegated task . For example, if and each product holons delegate a single task to product-ants, this implies 200 exploration results will be required for each evaluation for a layout change.

[Figure 6 near here]

#### Formal model of resource-ant’s state machine in SMS4MP

Each resource holon creates one resource-ant each, and constantly synchronises the state and not the location of the physical resources with the resource-ant. As shown in Fig. [7](#_bookmark9), the resource-ant randomly searches for product-ants and execute operations for product-ants in its *operation-range*, which is a range at which a resource-ant can sense a product-ant and vice versa. This is referred to as an exploration strategy. After each successful operation, the resource-ant stores its present location (referred to as *hotspot*) in the vector in memory and deposits a pheromone. When the location is full, the resource-ant exploits the *hotspots* stored in the vector before resulting to exploration strategy - this is referred to as exploitation strategy. The length of the vector determines how many previous locations the resource-ant can exploit before resulting to random exploration. While exploring the environment, if a new *hotspot* is found, the oldest *hotspot* is replaced with the new hotspot before restarting the exploitation procedure.

[Figure 7 near here]

The resource-ant’s exploitation strategy is complementary to the exploration strategy of the product-ant, which facilitates the discovery of optimal resource layout in the environment. For example, the product-ants deposit pheromones as they leave the *hotspot*, when the pheromones are discovered by another product-ants, the pheromones are reinforced, and more pheromones are deposited only if an operation is executed at the *hotspots* where the pheromones lead. Therefore, the exploitation strategy of the resource-ant ensures the reinforcement of pheromones for the product-ants. The length of the vector impacts the quality of the solution obtained by the product-ant and therefore have to be selected experimentally. If is too large, the resource-ants tend to exploit the environment more and explore less, therefore posing the danger of being trapped in a local minimum (over-fitting). When is too small, the resource-ants tend to explore more and exploit less, therefore leading to poor convergence (under-fitting).

#### Formal model of resource-bee’s state machine in SMS4MP

The resource-bee uses the presence of operations activities as its stigmergic coordination strategy. As shown in Fig. [8](#_bookmark10), the resource holons create resource-bees which explore the environment using a random strategy. However, when a product-ant is within *operation-range*, it executes an operation for the product-ant, provided the resource-bee is capable of executing the required operation. The resource-bee immediately compute a waiting time *tw* and wait at the present location called the *active-spot*. If and no product-ant with compatible operation need is found within its *operation-range*, the resource-bee leaves the *active-spot* and randomly explores the environment.

However, if a product-ant with compatible operation need is found, the resource-bee resets its waiting time to , where is the new waiting time, is the remaining waiting time, and is the value that determines the extra amount of time the resource-bee is required to wait at the same location.

[Figure 8 near here]

# Simulation and Results

The simulation is designed first, to show the capability of the system to autonomously execute layout changes in real-time to minimise the cost of executing personalisation processes in a highly flexible manufacturing system. Secondly, to evaluate the difference in performance when the resource-agent’s stigmergic behavioural strategy is based on the behaviour of ants or young honeybees. Two separate simulations are developed, the physical layer and the architecture are the same and only differ in the type of stigmergic strategies used by the resource-agents in the virtual layer. The first system uses the stigmergic behaviour inspired by ants for both the product and resources in the virtual layer, therefore, referred to as ant-inspired system (AIS). The second uses stigmergic behaviour inspired by ant for the products and stigmergic behaviour inspired by honeybee for the resources, and referred to as bee-inspired system (BIS).

A total of six production machines (resources) are used. Generally, for 6 production machines, there are 720 (6!) possible routing sequence. However, in the personalisation scenario used to evaluate the proposed technique, the number of possible routing sequence is assumed to be 14 (product groups) as shown in Table [2](#_bookmark11), and each resource in the system is assumed to be capable of performing 5 distinct operations. This implies that for a particular routing sequence, there will be a maximum of 120 possible operations sequence (unique product units). This gives a total of 1680 possible product units. During the simulation, the process of selecting a product within a group is random, while a simple tabu-list is used to ensure that no two products with equivalent operations sequence are selected within a group.

[Table 2 near here]

Table [3](#_bookmark12) shows the simulation dynamics. The horizontal-flow ∆*T*1, ∆*T*2, ∆*T*3, and ∆*T*4 represent stages in the simulation where the flow/ distribution of the different *product-groups* changes within the product mix. Each change in the horizontal-flow is equivalent to 5*k* simulation steps. The vertical flow represents stages in the simulation where the type of product group that constitutes the product mix changes; this is referred to as changes in product group ∆*PG*. Each of these changes leads to changes in the flow between the production resources, hence changes in material handling cost. There are a total of 20 changes in total and the flow-matrix representing the flow cost is presented in (Ogunsakin, 2020).

The flow matrix is integrated into the simulation dynamics, and as the system progresses from to , the different matrices as shown in Table [3](#_bookmark12) are introduced automatically. These changes are expected to be detected by the system, and stimulate changes in the system’s layout. The simulation is run for 125*k* simulation steps, 20*k* for each of the period ∆*PG*1 to ∆*PG*5, with a repetition of the ∆*PG*1 lasting for additional 20*k* simulation-steps. Additionally, the last epoch (∆*PG*1,∆*T*4) is run for additional 5*k* to further observe the self-organising capability of the system.

[Table 3 near here]

### Parameter selection

Table [4](#_bookmark13) shows the summary of parameter values for the experiment. The coefficient *ρ*, which determines how fast the deposited trail (pheromone) disappears from the virtual system is selected by evaluating several *ρ* values ranging from 0.001 to 0.1. The value *ρ* = 0.045 is observed to be the value of *ρ* for best convergence when the number *m* of *product-ants* in the virtual layer is set to 120. The number of robots (products and material handling systems) in the work in progress (WIP) of the real system at simulation time is set to 10. This implies that each robot delegates exploration task to 12 *product-ants* in the virtual layer. The number of available production machines is set to 6 in the physical layer, which is equivalent to the parameter *n* in the virtual layer. The default speed of product-ants is set to 0.7; this value is changed automatically based on the flow cost between resources. The speed of resource-ant is set to in the virtual and physical layer.

[Table 4 near here]

Two parameters and governed the stigmergic behaviour of the *resource-ants* and *resource-bees* respectively. The length represents the number of *hot-spots* each resource-ant stores in its memory. If is high, resource-ants tend to exploit stored location leading to over-fitting and poor performance. A low value of on the other hand signifies more exploration and less exploitation, leading to poor performance. The value of is selected experimentally by using values ranging from , the value gives the highest throughput. The parameter represents the waiting time, in simulation-time, for each resource-bee at the *active-spot*. The waiting time is selected experimentally to achieve a balance between exploration and exploitation of the environment – is set to .

The parameters and determines the sensitivity of the system to perturbation, and are therefore very important. The threshold parameter determines when the minimum cost should be augmented to guide the system out of a local minimum. A lower value of signifies frequent layout changes and a higher value of signifies less frequent layout changes. The values 30, 60, 90, and 120 are assigned to for both the ant-inspired and bee-inspired systems. The bee-inspired system shows a higher sensitivity to the value of due to the greediness of the stigmergic behaviour, as will be shown in a later section. In contrast, the ant-inspired system is comparatively less sensitive. Therefore, the values 90 and 60 are assigned to in the AIS and BIS respectively. The value of , which determines the rate at which the cost function is augmented, is set to 1 for both systems. This implies that the system takes a unit step in its attempt to escape from local minimum until a new optimal cost is obtained.

### Experimental Results

* + 1. *SMS4MP’s Self-organisation and sensitivity to perturbation*

The result of the simulation shows the effectiveness of the stigmergic strategies in reacting in real-time to constant changes in order-mix. Fig. 9 and Fig. 1[0](#_bookmark15) shows the dynamics of the optimisation process for the BIS and AIS respectively. The value of the threshold , which determines a candidate solution for the augmented cost function is set at 30, 60, 90, 120 in Fig. ([9](#_bookmark14) a, b, c, d) and Fig. [10](#_bookmark15) (a, b, c, d) respectively. It is observed that the system is able to achieve different optimal costs by using the augmented cost function to escape from local minimum. Each optimal cost achieved in the virtual layer is translated into a layout in the physical layer. Therefore, the rate at which the system achieves new optimal cost is equivalent to the rate of layout changes in the physical layer.

The sensitivity of the system to changes in order-mix reduces as the threshold increases. A high sensitivity implies frequent layout changes resulting from little changes in the system. Therefore, when = 30, the systems are observed to iterate over several optimal costs resulting in frequent layout changes. The number of the different optimal costs achieved is reduced as the threshold value increases, with = 120 resulting in the least number of layout changes.

As shown in Fig. 9 and Fig. 10. The BIS is more sensitive to the value of , which demonstrates higher sensitivity to changes in the environment. For example, the BIS records an average of 55, 32, 21, and 15 layout changes when = 30, 60, 90 and 12 respectively, while the AIS records an average of 32, 20, 10 and 5 layout changes when = 30, 60, 90, and 120 respectively. A balance is therefore required in the selection of an appropriate value for in the system, due to the disruptive impact of frequent layout changes on the system. Therefore, for the implemented system, the value of = 90 and = 60 is selected for the BIS and AIS respectively.

[Figure 9 near here]

[Figure 10 near here]

### SMS4MP’s Layout optimisation Capability: virtual layer

The systems are observed to possess the capability to minimise material flow cost by constantly optimising their respective layouts using stigmergic mechanism and an accompanying cost function that represents the individual experiences of the software agents in the virtual layer. The desired model is a model that is highly sensitive to changes in the manufacturing environment, but as a consequence, results in less number of layout changes.

The BIS, as shown in Fig. [11](#_bookmark16) outputs a total of 22 optimal costs at = 90; however, only 18 layout changes are executed in the physical layer. This is due to the occurrence of four concurrent events [(*j*, *k*), (*l*, *m*),(*n*, *o*), (*s*, *t*)] during the optimisation process, where two optimal costs are obtained at an interval smaller than the time required to complete one layout change. When this occurs, the least cost is taken, and the corresponding layout is executed. The system completes the execution of the first layout change if the first cost is lower than the second cost, or abandon the execution of the first to execute the second, if the second cost is lower than the first cost. The layouts corresponding to the different optimal costs are presented in Appendix A.

The AIS, as shown in Fig. [12](#_bookmark17) records a total of 19 optimal costs despite the value of = 60, with no instance of concurrency. All optimal costs with the corresponding layout are executed in the physical layer. The corresponding layouts for each optimal cost obtained are shown in Appendix B. Therefore, a comparison of the two systems shows that the BIS is highly sensitive to changes in product-mix and make fewer layout changes to its layout, representing a better system compared to the AIS. However, sensitivity does not equal high throughput or lower average cycle-time in the physical layer. Therefore, evaluating the cycle-time for each model is essential to justify the effectiveness of the BIS over the AIS.

[Figure 11 near here]

[Figure 12 near here]

### SMS4MP’s Layout optimisation Capability: physical layer

The cycle-time, which is the average time taken to manufacture each product, is used to evaluate the effectiveness of the optimisation strategies employed by the two systems. The simulation is run for a total of 125*k* simulation steps, and the cycle-time is calculated at each 1*k* simulation steps. 20 different simulation runs were performed for each system, and an average cycle-time is computed, as shown in Fig. [13](#_bookmark18).

To justify the effectiveness of the system to frequent changes in product mix, and to ensure that the system is not performing generally worse than expected, the different resources are placed on a straight line (single row) and rectangular shape (multi-rows) with an overlapping constraint of 10, which represents the minimum distance between the resources (as used in the simulation). The layout for the single and multi-row static layout configurations are estimated using the IDFMs (Ogunsakin, 2020). For the two layout configurations, the least average cycle-time obtained for 20 simulation runs without layout changes is 463.25. Therefore, the cycle-time for the BIS and AIS are expected to be within the range of the average cycle-time of 463.25. Also, a maximum cycle-time of 948 is obtained in the static layout by separating each resource at a maximum possible distance. Therefore, at the start of each of the simulation for the BIS and AIS, the resources are placed at locations where they are separated by maximum possible distances. It should be noted that the estimated cycle-time for the static facility layouts are for control purposes, and not to serve as a comparison to the proposed system.

The BIS is shown to immediately achieve an average cycle-time lower than 400 at first iteration with the majority of the cycle-time obtained ranging between 80 and 145 as shown in Fig.[13](#_bookmark18)(a) compare to the AIS. The AIS gradually iterates from *approx*. 600, with the majority of the cycle-time having a value above 120, as shown in Fig. [13](#_bookmark18)(b). The BIS is also observed to be more stable compared to the AIS.

Fig.14 shows the differences in the cycle-time. In Fig. [14](#_bookmark19)(a), the cycle-time reduces progressively from simulation-time *t* = 0*k* to *t* = 20*k*. From *t* = 20*k*, which is the beginning of ∆*PG*2. When a new product group is introduced into the system (see Table [3](#_bookmark12)), the cycle-time starts to increase. However, the system quickly recovers from this perturbation by consistently minimising the cycle-time from *t* = 30*k* simulation step through *t* = 60*k*. No serious impact is observed at *t* = 40*k* when 2 new product-groups are introduced as the distribution of the different group is kept constant during this period (from ∆*PG*2,∆*T*4 to ∆*PG*3,∆*T*1). From *t* = 60*k* through *t* = 75*k*, the cycle-time is observed to increase as 3 new product-groups are introduced, and the distribution is changed at the same time. Finally, at *t* = 80*k* through *t* = 105*k*, when the four product-groups are replaced by four new product-groups, the system experiences maximum perturbation at this point, but gradually recovers as observed in the reduction in cycle-`time, and despite further changes from *t* = 105*k* through *t* = 125*k*, a reduction in cycle-time is observed.

[Figure 13 near here]

[Figure 14 near here]

Despite the constant perturbation as a result of changes in product distribution within the product mix, the introduction of new product groups, and constantly changing product types, the system consistently attempts to minimise the cycle-time by constantly evolving the layout to cope with these perturbations. Fig. [14](#_bookmark19)(b) shows that 75% of the cycle-time for the BIS is below 145, while 75% of the cycle-time for the AIS is above 155. This shows the benefit of using the BIS over the AIS.

# Discussion and Conclusion

The goal of mass personalisation is to satisfy the needs of individual customers by incorporating their input into the design phase of the product, and to produce the resulting personalised products at a cost comparable to that of mass production (Pine, [1993](#_bookmark60); Koren et al., [2015](#_bookmark46)). Approaches to address unpredictable changes in manufacturing systems have been largely based on the heterarchical structure of control and autonomous, flexible and intelligent manufacturing entities (Pujo et al., [2009](#_bookmark61); Leita˜o & Restivo, [2006](#_bookmark49); Barbosa et al., [2015](#_bookmark20); Kumar, [2007](#_bookmark47); Mourtzis & Doukas, [2014](#_bookmark51)). However, all these paradigms are constrained by the rigid architecture of the manufacturing systems for which they are implemented, such that the flexibility and autonomy achieved in the digital space are not fully realised in the physical systems.

The approach proposed in this paper takes inspiration from the research in the manufacturing industries (Bosch, [2016](#_bookmark26); Rexroth, [2018](#_bookmark64); Kuka, [2019](#_bookmark0)). The approach relaxes the rigid constraints imposed on the present manufacturing systems through the introduction of mobile production machines, and replaces conveyor systems with mobile intelligent robots for material handling. These changes ensure that the physical system can realise the flexibility achievable in the digital space through the use of an ant-based and bee-based heuristics.

The SMS4MP is shown to evolve its layout in real-time through the use of stigmergic coordination mechanism, keeping up with product changes. This capability is observed in the autonomous layout changes and autonomous routing of products in the system (see Appendix C). The virtual layer uses stigmergy as the coordination mechanism, and through local interactions between the software agents, the overall layout of the system emerges in real-time. However, due to the stochastic order arrival time and product type, the system has to exhibit organisation closure, where after attaining a stable and optimal configuration, it constantly creates and delegate software agents in the virtual layer to explore new optimal configuration to maintain the present optimal state as shown in Fig.13.

Two different, yet complementary stigmergic approaches are used in SMS4MP. The product-ants behave like the typical ants, while the resource-ants instead use private pheromones to remember previous *hot-spots*. This abstraction proves to be effective but less effective compared to the model where resources are represented as young honeybees, called resource-bees. The resource-bees use temporal features of the environment, which is the presence of corresponding operation as a stigmergic variable(Schmickl & Hamann, [2010](#_bookmark67)).

The use of compatible manufacturing operations by the bee-inspired system as stigmergic variables and exploitation mechanism, provide a better fit for the problem. Thus, the bee-inspired system emerges as the most effective approach. Also, this shows that a combination of multiple and complementary nature-inspired models like ant-based optimisation algorithm (Dorigo & Di Caro, [1999](#_bookmark34)) and beeclust algorithm (Schmickl & Hamann, [2010](#_bookmark67)) are beneficial when the problem space cannot be modelled using a single natural model as observed in this research. Finally, the system shows the capability of executing personalisation process without excessive increase in cost, irrespective of the resulting stochastic order arrival and heterogeneous order types.

### Conclusion

This research paper proposes and implements a self-organising manufacturing system for mass personalisation through the introduction of mobile production resources, replacement of static conveyor system with mobile and intelligent robot for material handling, application of appropriate stigmergic mechanism as observed in the colonies of ants and bees, and the use of delegate multi-agent system as the coordination mechanism between the virtual layer populated by software agents and the physical layer populated by physical manufacturing equipment. The experimental evaluation of SMS4MP demonstrate considerable contribution to the design and engineering of manufacturing systems for mass personalisation, the use of stigmergic strategy in manufacturing control, and the method and process of engineering self-organising manufacturing system in mass personalisation context.

Further research is needed to demonstrate the viability of the proposed system using a prototype manufacturing system. However, most of the available tools and software for fast prototyping of manufacturing systems are strongly based on fixed production machines and rigid conveyor systems. Therefore, the only possibility is to build a prototype from the ground up. However, before this, a strong theoretical foundation is required to demonstrate the possibility and viability of the approach using formal and computational tools, which is presented in this research. Hence, this paper presents such a foundation on which a prototype system can be built for further research in academia and industry.

**References**

Barbosa, J. & Leitao, P. (2011). Simulation of multi-agent manufacturing systems using agentbased modelling platforms. In *IEEE international conference on industrial informatics (indin)*(pp. 477–482). doi:[10.1109/INDIN.2011.6034926](https://dx.doi.org/10.1109/INDIN.2011.6034926)

Barbosa, J., Leita˜o, P., Adam, E., & Trentesaux, D. (2015). Dynamic self-organization in holonic multi-agent manufacturing systems: The ADACOR evolution. *Computers in Industry*, *66*, 99–111. doi:[10.1016/j.compind.2014.10.011](https://dx.doi.org/10.1016/j.compind.2014.10.011)

Bodi, M., Thenius, R., Szopek, M., Schmickl, T., & Crailsheim, K. (2012). Interaction of robot swarms using the honeybee-inspired control algorithm BEECLUST. *Mathematical and Computer Modelling of Dynamical Systems*, *18*(1), 87–100. doi:[10.1080/13873954.2011.601420](https://dx.doi.org/10.1080/13873954.2011.601420)

Boe¨r, C. R., Dulio, S., & Jovane, F. (2004). Editorial: Shoe design and manufacturing. *International* *Journal of Computer Integrated Manufacturing*, *17*(7), 577–582. doi:[10.1080/09511920412331292637](https://dx.doi.org/10.1080/09511920412331292637)

Bongaerts, L., Monostori, L., McFarlane, D., & Ka´da´r, B. (2000). Hierarchy in distributed shop floor control. *Computers in Industry*, *43*(2), 123–137. doi:[10.1016/S0166-3615(00)00062-2](https://dx.doi.org/10.1016/S0166-3615%2800%2900062-2)

BoschGmbh(2016). Press release: Bosch is working with international research partners to develop a new modular manufacturing system.

Botti, V. & Adriana Giret. (1997). Holonic Manufacturing Systems. *A multi-agent methodology for Holonic Manufacturing*, 432–438. doi:[10.1007/978-1-84800-310-1](https://dx.doi.org/10.1007/978-1-84800-310-1)

Delloite. (2019). *The Deloitte Consumer REview: Made-to-order: The rise of mass personalisation Contents*.

Di Caro, G. & Dorigo, M. (1998). AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, *9*, 317–365. doi:[10.1613/jair.530](https://dx.doi.org/10.1613/jair.530). arXiv: [1105.5449](http://arxiv.org/abs/1105.5449)

Di Marzo Serugendo, G., Foukia, N., Hassas, S., Karageorgos, A., Moste´faoui, S. K., Rana, O. F.,

 Van Aart, C. (2004). Self-organisation: Paradigms and applications. In *Lecture notes in* *artificial intelligence (subseries of lecture notes in computer science)* (Vol. 2977, pp. 1–19).

Di Marzo Serugendo, G., Gleizes, M. P., & Karageorgos, A. (2011). Self-organising Software: From Natural to Artificial Adaptation. *Natural Computing Series*, *37*. doi:[642-17348-6](https://dx.doi.org/10.1007/978-3-642-17348-6)

Di Marzo Serugendo, G., Gleizes, M. P., & Karageorgos, A. (2006). Self-organisation and emergence in MAS: An overview. *Informatica (Ljubljana)*, *30*(1), 45–54.

Di Roma, A. (2017). Footwear Design. The paradox of ”tailored shoe” in the contemporary digital manufacturing systems. *The Design Journal*, *20*(sup1), S2689–S2699. doi:[10.1080/](https://dx.doi.org/10.1080/14606925.2017.1352780) [14606925.2017.1352780](https://dx.doi.org/10.1080/14606925.2017.1352780)

Dorigo, M. & Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, *2*, 1470–1477. doi:[10.1109/CEC.1999.782657](https://dx.doi.org/10.1109/CEC.1999.782657)

Dorigo, M., Bonabeau, E., & Theraulaz, G. (2000). Ant algorithms and stigmergy. *Future Gener* *ation Computer Systems*, *16*(8), 851–871. doi:[10.1016/S0167-739X(00)00042-X](https://dx.doi.org/10.1016/S0167-739X%2800%2900042-X)

Gao, Q., Luo, X., & Yang, S. (2005). Stigmergic cooperation mechanism for shop floor con trol system. *International Journal of Advanced Manufacturing Technology*, *25*(7-8), 743–753. doi:[10.1007/s00170-003-1901-x](https://dx.doi.org/10.1007/s00170-003-1901-x)

Grasse¨, P. (1959). La reconstruction du nid et les coordinations InterIndividuelles chez Bellicositermes natalensis et Cubitermes sp. la the´orie de la stigmergie: Essai d’interpre´tation du comportement des termites constructeurs. *Social Insect*, *6*, 41–80.

Hani, Y., Amodeo, L., Yalaoui, F., & Chen, H. (2007). Ant colony optimization for solving an industrial layout problem. *European Journal of Operational Research*, *183*(2), 633–642. doi:[10.](https://dx.doi.org/10.1016/j.ejor.2006.10.032) [1016/j.ejor.2006.10.032](https://dx.doi.org/10.1016/j.ejor.2006.10.032)

Heredia, D. A., Ceballos, F., & Sanchez-Torres, G. (2018). Simulation-Based Improvement Procedure for Small-Scale Shoe Manufacturing Companies. *Journal of Advanced Manufacturing Systems*, *17*(1), 23–33. doi:[10.1142/S0219686718500026](https://dx.doi.org/10.1142/S0219686718500026)

Hevner, A. R., March, S. T., Jinsoo Park, & Ram, S. (2004). Design Science in Information System Research. MIS Quarterly, 28(1), 75–105.

Heylighen, F. (1989). Self-organization, emergence and the architecture of complexity. *Complexity*, *18*, 23–32. Retrieved from [http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.7389%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf) [1.1.32.7389%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.7389%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf)

Heylighen, F. (2001). The science of self-organization and adaptativity. *The Encyclopedia of Life* *Support Systems*, 1–26. doi:[10.1.1.38.7158](https://dx.doi.org/10.1.1.38.7158)

Heylighen, F. (2016). Stigmergy as a universal coordination mechanism I: Definition and components. *Cognitive Systems Research*, *38*, 4–13. doi:[10.1016/j.cogsys.2015.12.002](https://dx.doi.org/10.1016/j.cogsys.2015.12.002)

Hsiao, W. P. & Chiu, M. C. (2014). A mass personalization methodology based on co-creation. In

*Advances in transdisciplinary engineering* (pp. 698–705). doi:[10.3233/978-1-61499-440-4-698](https://dx.doi.org/10.3233/978-1-61499-440-4-698) Hu, S. J. (2013). Evolving paradigms of manufacturing: From mass production to mass customization and personalization. In *Procedia cirp* (Vol. 7, pp. 3–8). Elsevier B.V. doi:[10.1016/](https://dx.doi.org/10.1016/j.procir.2013.05.002)

[j.procir.2013.05.002](https://dx.doi.org/10.1016/j.procir.2013.05.002)

Komarudin & Wong, K. Y. (2010). Applying Ant System for solving Unequal Area Facility Layout Problems. *European Journal of Operational Research*, *202*(3), 730–746. doi:[10.1016/j.ejor.](https://dx.doi.org/10.1016/j.ejor.2009.06.016) [2009.06.016](https://dx.doi.org/10.1016/j.ejor.2009.06.016)

Koren, Y., Shpitalni, M., Gu, P., & Hu, S. J. (2015). Product design for mass-individualization.

*Procedia CIRP*, *36*, 64–71. doi:[10.1016/j.procir.2015.03.050](https://dx.doi.org/10.1016/j.procir.2015.03.050)

Kumar, A. (2007). From mass customization to mass personalization: A strategic transformation. *International Journal of Flexible Manufacturing Systems*, *19*(4), 533–547. doi:[10 . 1007 /](https://dx.doi.org/10.1007/s10696-008-9048-6) [s10696-008-9048-6](https://dx.doi.org/10.1007/s10696-008-9048-6)

Leita˜o, 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.](https://dx.doi.org/10.1016/j.engappai.2008.09.005) [09.005](https://dx.doi.org/10.1016/j.engappai.2008.09.005)

Leita˜o, P. & Restivo, F. (2006). ADACOR: A holonic architecture for agile and adaptive manufacturing control. *Computers in Industry*, *57*(2), 121–130. doi:[10.1016/j.compind.2005.05.005](https://dx.doi.org/10.1016/j.compind.2005.05.005)

Mondada, F., Pettinaro, G. C., Guignard, A., Kwee, I. W., Floreano, D., Deneubourg, J. L., . . . Dorigo, M. (2004). Swarm-bot: A new distributed robotic concept. *Autonomous Robots*, *17*(2-3), 193–221. doi:10.1023/B:AURO.0000033972.50769.1c

Mourtzis, D. & Doukas, M. (2014). Design and planning of manufacturing networks for mass customisation and personalisation: Challenges and outlook. In *Procedia cirp* (Vol. 19, *100*, pp. 1–13). Elsevier B.V. doi:[10.1016/j.procir.2014.05.004](https://dx.doi.org/10.1016/j.procir.2014.05.004)

Nouiri, M., Bekrar, A., Jemai, A., Niar, S., & Ammari, A. C. (2018). An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing*, *29*(3), 603–615. doi:[10.1007/s10845-015-1039-3](https://dx.doi.org/10.1007/s10845-015-1039-3)

Ogunsakin, R., Mar´ın, C. A., & Mehandjiev, N. (2018). Self-Reconfigurable Manufacturing System For Personalized Mass Customisation. (100), 1–8.

Ogunsakin, R., Mehandjiev, N., & Mar´ın, C. A. (2018). Bee-inspired self-organizing flexible manufacturing system for mass personalization. In P. Manoonpong, J. C. Larsen, X. Xiong,

J. Hallam, & J. Triesch (Eds.), *From animals to animats 15* (pp. 250–264). Cham: Springer International Publishing.

Omicini, A. (2013). Nature-Inspired Coordination Models: Current Status and Future Trends.

*ISRN Software Engineering*, *2013*, 1–13. doi:[10.1155/2013/384903](https://dx.doi.org/10.1155/2013/384903)

Omicini, A. & Viroli, M. (2011). Coordination models and languages: From parallel computing

to self-organisation. *Knowledge Engineering Review*, *26*(1), 53–59. doi:[10.1017/S026988891000041X](https://dx.doi.org/10.1017/S026988891000041X) Onori, M., Lohse, N., Barata, J., & Hanisch, C. (2012). The IDEAS project: plug & produce at

shop-floor level. *Assembly Automation*, *32*(2), 124–134. doi:[10.1108/01445151211212280](https://dx.doi.org/10.1108/01445151211212280) Pach, C., Berger, T., Bonte, T., & Trentesaux, D. (2014). ORCA-FMS: A dynamic architecture

for the optimized and reactive control of flexible manufacturing scheduling. *Computers in* *Industry*, *65*(4), 706–720. doi:[10.1016/j.compind.2014.02.005](https://dx.doi.org/10.1016/j.compind.2014.02.005)

Peeters, P., Van Brussel, H., Valckenaers, P., Wyns, J., Bongaerts, L., Kollingbaum, M., & Heikkila¨,

T. (2001). Pheromone based emergent shop floor control system for flexible flow shops.

*Artificial Intelligence in Engineering*, *15*(4), 343–352. doi:[10.1016/S0954-1810(01)00026-7](https://dx.doi.org/10.1016/S0954-1810%2801%2900026-7) Pine, J. B. (1993). Mass customization: The new frontier in business competition. *Harvard Busi-*

*ness School Press.*

Pujo, P., Broissin, N., & Ounnar, F. (2009). PROSIS: An isoarchic structure for HMS control. *Engineering Applications of Artificial Intelligence*, *22*(7), 1034–1045. doi:[10.1016/j.engappai.](https://dx.doi.org/10.1016/j.engappai.2009.01.011) [2009.01.011](https://dx.doi.org/10.1016/j.engappai.2009.01.011)

Pujo, P., Dubromelle, Y., & Ounnar, F. (2010). RFID uses for prosis ambient control. In *Proceedings of the 4th international workshop on rfid technology concepts, applications, challenges, iwrt* *2010, in conjunction with iceis 2010* (Iceis, pp. 99–106). doi:[10.5220/0002911000810088](https://dx.doi.org/10.5220/0002911000810088)

Raj, J. A., Ravindran, D., Saravanan, M., & Prabaharan, T. (2014). Simultaneous scheduling of machines and tools in multimachine flexible manufacturing systems using artificial immune system algorithm. *International Journal of Computer Integrated Manufacturing*, *27*(5), 401–414. doi:[10.1080/0951192X.2013.834461](https://dx.doi.org/10.1080/0951192X.2013.834461)

Rexroth, B. (2018). Hm18 factory of the future show. Retrieved from [https://www.youtube.](https://www.youtube.com/watch?v=HNK7By5W0e8) [com/watch?v=HNK7By5W0e8](https://www.youtube.com/watch?v=HNK7By5W0e8). ((accessed: 04.09.2019))

Sallez, Y., Berger, T., & Tahon, C. (2004). Simulating intelligent routing in flexible manufacturing systems using NetLogo. In *Proceedings of the ieee international conference on industrial* *technology* (Vol. 2, pp. 1072–1077).

Sallez, Y., Berger, T., & Trentesaux, D. (2009). A stigmergic approach for dynamic routing of active products in FMS. *Computers in Industry*, *60*(3), 204–216. doi:[10 . 1016 / j . compind.](https://dx.doi.org/10.1016/j.compind.2008.12.002) [2008.12.002](https://dx.doi.org/10.1016/j.compind.2008.12.002)

Schmickl, T. & Hamann, H. (2010). BEECLUST: A swarm algorithm derived from honeybees.

*Bio-inspired Computing and Networking*, 95–137.Solimanpur, M., Vrat, P., & Shankar, R. (2004). Ant colony optimization algorithm to the intercell layout problem in cellular manufacturing. *European Journal of Operational Research*, *157*(3), 592–606. doi:[10.1016/S0377-2217(03)00248-0](https://dx.doi.org/10.1016/S0377-2217%2803%2900248-0)

Song, H. S. & Kim, T. G. (1994). The DEVS framework for discrete event systems control. In *Proceedings of the 5th annual conference on ai, simulation, and planning in high autonomy systems: Distributed interactive simulation environments, aihas 1994* (pp.228–234). doi:[10.1109/](https://dx.doi.org/10.1109/AIHAS.1994.390484) [AIHAS.1994.390484](https://dx.doi.org/10.1109/AIHAS.1994.390484)

Tseng, M. M., Jiao, R. J., & Wang, C. (2010). Design for mass personalization. *CIRP Annals*  *Manufacturing Technology*, *59*(1), 175–178. doi:[10.1016/j.cirp.2010.03.097](https://dx.doi.org/10.1016/j.cirp.2010.03.097)

Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., & Peeters, P. (1998). Reference architecture for holonic manufacturing systems: PROSA. *Computers in Industry*, *37*(3), 255–274. doi:[10.1016/S0166-3615(98)00102-X](https://dx.doi.org/10.1016/S0166-3615%2898%2900102-X)

Verstraete, P., Saint Germain, B., Valckenaers, P., Van Brussel, H., Van Belle, J., Karuna, H., . . . Karuna, H. (2008). Engineering manufacturing control systems using PROSA and delegate MAS. *International Journal of Agent-Oriented Software Engineering*, *2*(1), 62–89. doi:[10.](https://dx.doi.org/10.1504/IJAOSE.2008.016800) [1504/IJAOSE.2008.016800](https://dx.doi.org/10.1504/IJAOSE.2008.016800)

Viroli, M., Casadei, M., & Omicini, A. (2009). A framework for modelling and implementing self-organising coordination. In *Proceedings of the acm symposium on applied computing*(pp. 1353–1360). doi:[10.1145/1529282.1529585](https://dx.doi.org/10.1145/1529282.1529585)

Wang, Y., Ma, H. S., Yang, J. H., & Wang, K. S. (2017). Industry 4.0: a way from mass customization to mass personalization production. *Advances in Manufacturing*, *5*(4), 311–320. doi:[10.1007/s40436-017-0204-7](https://dx.doi.org/10.1007/s40436-017-0204-7)

Weyns, D., Schumacher, M., Ricci, A., Viroli, M., & Holvoet, T. (2005). Environments in multiagent systems. *Knowledge Engineering Review*, *20*(2), 127–141. doi:[10.1017/S0269888905000457](https://dx.doi.org/10.1017/S0269888905000457)

Wolpert, D. H. (2003). Collective intelligence. In *Computational intelligence: The experts speak*

(pp. 245–260). doi:[10.1109/9780470544297.ch17](https://dx.doi.org/10.1109/9780470544297.ch17)

Zacharewicz, G. (2008). G-DEVS / HLA Environment for Distributed Simulations of Workflows. *84*(5), 197–213. doi:[10.1177/0037549708092833](https://dx.doi.org/10.1177/0037549708092833)

Ogunsakin, R. (2020, October 28). IDFM\_DATA.xlsx. figshare. Retrieved October 28, 2020, from https://figshare.com/articles/dataset/IDFM\_DATA\_xlsx/13153619/1