A Core Reference Model for Applicable Reconfigurable Manufacturing Systems

Pascal André, Olivier Cardin

A Core Reference Model for Applicable Reconfigurable Manufacturing Systems

Pascal André and Olivier Cardin

Abstract Reconfigurable Manufacturing Systems (RMS) have emerged as a promising approach to address the dynamic demands and uncertainties in modern manufacturing. However, the lack of a comprehensive and universally accepted definition of the notion of configuration in RMS poses challenges for researchers and practitioners. This research article aims to bridge this gap by suggesting a core reference model for RMS, allowing to exhibit a generic modelling of configurations. The article critically examines existing definitions and approaches to configuration, providing a comprehensive overview of the different dimensions and components of configuration in RMS. By addressing the lack of a consistent understanding of configuration in RMS, this article contributes to the development of a shared knowledge base and paves the way for further advancements in the field. It also highlights the importance of establishing a clear and holistic definition of configuration for effective implementation and utilisation of RMS in practice.

Key words: Reconfigurable Manufacturing Systems, Metamodel, Holon, Concerns, Domain Specific languages

1 Introduction

In the face of rapidly changing market demands, increasing product complexity, and technological advancements, manufacturers are seeking innovative solutions to enhance their agility, adaptability, and competitiveness. Reconfigurable Manufacturing Systems (RMS) have emerged as a promising approach to meet these challenges by providing a flexible and responsive manufacturing environment. However, despite the growing interest and adoption of RMS, there remains a critical gap in the understanding and definition of the notion of configuration within these systems. Con-

Pascal André · Olivier Cardin
LS2N UMR CNRS 6004 - University of Nantes and IUT de Nantes – 2, rue de la Houssinière F-44322 Nantes Cedex, France, e-mail: name.surname@ls2n.fr

figuration, as a fundamental concept in RMS, encompasses the arrangement, organization, and adaptation of system components, processes, and resources to meet specific manufacturing requirements. It encompasses the ability to modify and reorganize the system's physical, logical, and operational attributes to accommodate changing product specifications, production volumes, and market conditions. The configuration of an RMS influences its adaptability, scalability, and overall performance, thereby playing a crucial role in achieving operational excellence and competitive advantage.

Currently, there is a lack of a comprehensive and universally accepted definition of configuration in the context of RMS. This gap hampers researchers and practitioners in effectively understanding, designing, and implementing RMS. The objective of this article is to bridge the existing gap by suggesting a core reference model for RMS. The ultimate objective is to establish a foundation for a unified and comprehensive definition of configuration in RMS. To do so, this article will examine the different aspects and components of configuration, targeting modular design, adaptability, flexibility, and reconfigurability.

The findings of this research aims at contributing to the development of a shared knowledge base for researchers and practitioners in the field of RMS. A clear and holistic understanding of the notion of configuration will enable more effective design, implementation, and utilization of RMS in practice. It will also facilitate the development of standardized methodologies, tools, and frameworks for configuring RMS to meet evolving manufacturing needs.

The primarily targeted methodologies are related to a research project (RODIC project, funded by the French National Research Agency - ANR) where the focus is specifically on the evaluation phase of a given potential configuration. The configuration can therefore be roughly defined as a set of modules, arranged in a given manner with given parameters. In this phase of evaluation, several actions have to be executed: (i) let the user define its configuration in a computable form (ii) verify the validity of the configuration (iii) define the test scenario that will be executed (iv) evaluate the performance indicators of the configuration in this scenario (v) present the indicators in an understandable form to the user. This list exhibits the need to have different perspectives on the notion of configuration all along the evaluation phase. Therefore, it seems valuable to connect all those perspectives with a single reference model, able to capture all the characteristics of a configuration in a RMS context.

The paper is organised as follows. The background information is given in Section 2. Section 3 expresses the main principles of applicable reference architectures which are put in practice in the reference model we propose in Section 4. The architecture is modular to be customized in various contexts of RMS and then illustrated in a simple case in Section 5. In conclusion, perspectives are drawn for a larger integration and development of the tool in an actual manufacturing context.

2 Background

Reconfigurable Manufacturing Systems were initially defined more than 20 years ago [6]. Along the years, many studies focused on: (i) the definition of the expectations of such a paradigm (scalability, changeability...) [20] and the way to measure these features [22], (ii) the definition of modules (often called machines) compatible with the RMS concept [17], (iii) the optimisation of the configuration definition [21]. This last item was extensively studied in an Operational Research perspective, reusing the computation tools that are classically dedicated to the optimisation of manufacturing line: layout design [13], line balancing [8], or planning/scheduling [7]. However, in all this field, the notion of configuration is reduced to its simplest expression, in order to reduce the complexity of the calculations. The corollary of this hypothesis is to simplify the structure of the targeted configurations, often as a variation of a flowshop. In this context, it becomes then impossible to consider any kind of intelligence in the control of the RMS.

The only possibility that is provided to the end user wanting to evaluate a complex configuration is the use of discrete-event simulation [1]. This kind of tool is well-known for the last 40 years, and proved its efficiency in both the ability to model complex configurations and to evaluate various performance indicators. However, the classical drawbacks of the approach are exacerbated in the context of RMS *i.e.* (i) the time pressure on the model definition process, with potentially several models to develop to compare various potential configurations, (ii) the time pressure on the simulation runs, as the classical trial-and-error methodology is often time-consuming, and (iii) the expected skills of the operator in charge of the reconfiguration decision, when these kinds of tools are generally used in the design phase [19]. Therefore, in a context of advanced manufacturing systems (*e.g.* in the field of intelligent manufacturing), there is a need for new design methodologies for performance evaluation, speeding up or automating some of the simulation model design phases. Hence the need for a generic modelling effort targeting the configuration of RMS as the basis of the development of this advanced design methodologies.

Some previous works have already started to define RMS models, often based on a UML modelling paradigm. For example, authors in [10] proposed several concurrent models, later used in both simulation and operational research studies, in order to compare different layouts of an industrial case. The different models were fully dedicated to the industrial case that was tackled during this PhD work. Therefore, they exhibit a certain lack of genericity, which is very common in RMS literature. Hence the need for a RMS reference model.

3 Requirements

This section discusses requirements for RMS reasoning and applicability. Most belong to the *Stream #2–analysis of RMS features* of Bortolini et al.'s survey [6]: modularity, integrability, diagnosibility, convertibility, customisation and scalability.

① **Conceptual models for RMS.** The survey of Kayser et al. [16] pointed out that detailed reference architectures are under-specified to be applicable, there miss detailed models to design the software applications of the manufacturing systems, leaving such decisions to software delivers. As mentioned in [6], conceptual models for RMS are a missing but promising research stream. In this article, we use the UML standard notation. To improve the modelling quality, we refer to the *core model principles* of [3]: meaningful notation, type/instance distinction, recursive aggregation, aggregation protocols, schematic specialisations.

② **Modularity.** Reconfiguration cannot operate on any random MS concept, there must be reconfiguration units. As mentioned in [11], modularity is a basic requirement for RMS and module-based configuration enables reconfiguration. *Indeed, elements can be put together either to adjust the production volume (scalability), or to add functionality to the system (convertibility), or to produce more customised products (customisation).*

③ **Type and Instance Models.** As mentioned in [3], in Manufacturing Execution System (MES) we need both type and instance at the modelling level because instances have several occurrences. A resource-type (*e.g.* an arm robot) describe capabilities. A resource-instance (*e.g.* FANUC LR Mate 200) defines a set of occurrences (instances), each having its customised information (id, values...). The same applies for products defined by product-types (*e.g.* with a nomenclature) and produced in several instances according to a product order. In Object Orientation, metaprotocol allow the three levels instantiation: an instance is defined by a class which is defined by a metaclass (right part of Fig. 1). In our case we will simply use class modelling (right part of Fig. 1): a module-type defines module that are instantiated for each occurrence.

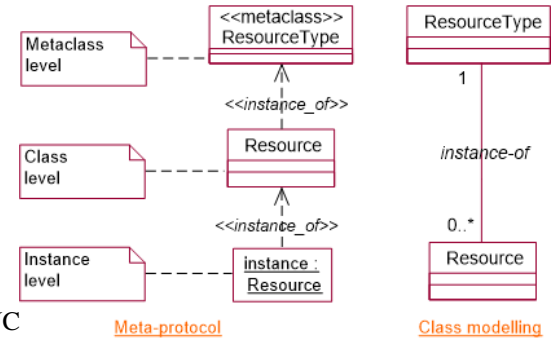

Fig. 1 Types vs instance models

④ **Abstraction and encapsulation.** Abstraction is a key feature in modular design. A configuration is an assembly of modules that focuses on the module interface (an abstraction) not the module implementation that encapsulates data, control... The interface must include the information to decide whether modules can be linked together or not *(integrability)*.

⑤ **Recursive aggregation.** A composite module encapsulates a configuration of other modules (its implementation). A composite module can be part of a bigger configuration (according to its interface). This is an elegant way to achieve *scalability* and improves part to part *diagnosability*. For example, a module can range from a single resource to a factory module.

⑥ **Multi-aspects and loose coupling.** A manufacturing system involves several stakeholders with various concerns: CPS engineers (physical modules), automation engineers (resources control), software engineers (communication networks and MES), business managers (KPI)... Fotso et al. [11] propose a fractal vision of modules where four aspects are associated to modules (physical, control, simulation, KPI). However, their (high-level) reference model is not applicable because the glue for integration (based on the production process) has not been studied and the viewpoints are not continuously fractal among recursive aggregation. Last, the aspects must be as independent as possible to be reusable (*loose coupling*).

⑦ **Early verification and strong consistency.** The consistent and complete integration of the viewpoints is the most important challenge to tackle for RMS applicability. In addition to model modularity, we need modelling language modularity by composing Domain Specific Languages (DSL) to face heterogeneity. System modelling consider three axes: structure (organisation), dynamic behaviour (or control) and functional behaviour (or actions) Smaller DSL improves *verifiability* and *diagnosability* by focusing on specific properties (*strong consistency*) *e.g.* deadlock freeness is associated to control not to the structure or computations.

4 RMS Reference Model

This section provides an excerpt of the reference architecture, that has been built upon the principles of Section 3. The models will be illustrated with expressive modelling notations (principle ①). Fig. 2 shows the general organisation of the reference architecture. The RMS_Modules describes modules and configurations including layouts and organisational units. The RMS_Control layer adds dynamic information to control modules. The RMS_Product layer describes the products. The RMS_Production layer adds functional information for the choreography and orches-

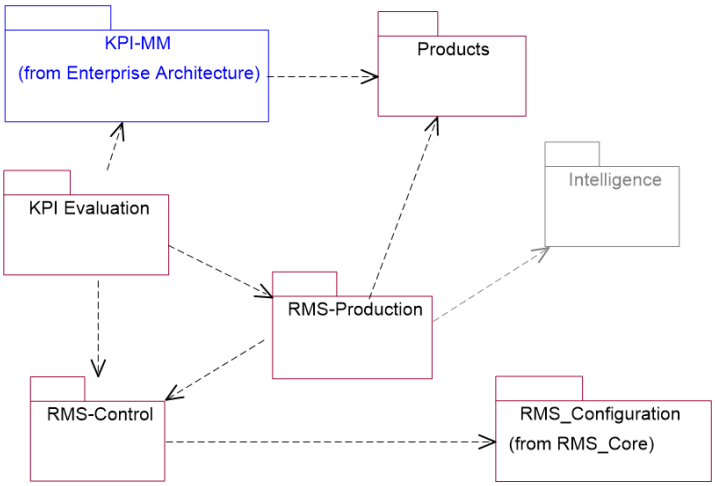

Fig. 2 Main Reference Architecture

tration of manufacturing. The Intelligence package adds facilities for scheduling and other ordering activities. The KPI–MM (metamodel) layer defines KPI, at the enterprise level basically on product information. The KPI are instrumented in the RMS by the KPI Evaluation integrating layer that provides evaluation means. Last the Simulation enables to compute metrics to evaluate the RMS before effective reconfiguration. Note that it is a logical model and the physical part is hidden under RMS_Architecture and RMS_Control. Next we focus on the main trends so the secondary concern Intelligence is not detailed.

4.1 Modules and Configurations

The core concepts of the RMS reference architecture are represented in Fig. 3. A **module** defines information (attributes or data objects) and operations (principle ② and ④). The available operations (*e.g.* the functional capabilities of a resource) are published in Interface concept. A Service is a high-level operation with an API (inner operations), service Contracts (QoS) and protocols (how to use the operations). The concept of EndPoint enables to connect modules through Binding s. An **assembly** is a set of modules connected by assemblyBindings on their end points: this is a client/server relation. For example, "*taking a product on a conveyor*" can be seen as binding the "leave" operation of a conveyor module with the "take" operation of a workstation module. However, operations as well as interfaces can be specialised as provided or required capabilities, this may depend on the implementation of the reference architecture. Behind the input parameters and output results, operations may exchange data through messages. Similarly to modules, AssemblyType is the abstraction of assemblies at the Module–type level (principle ③). CompositeModule enables scalability by encapsulating an assembly of modules (principle ⑤-recursion) and by being perceived as a (black-box) module of the assembly that contains it (prin-

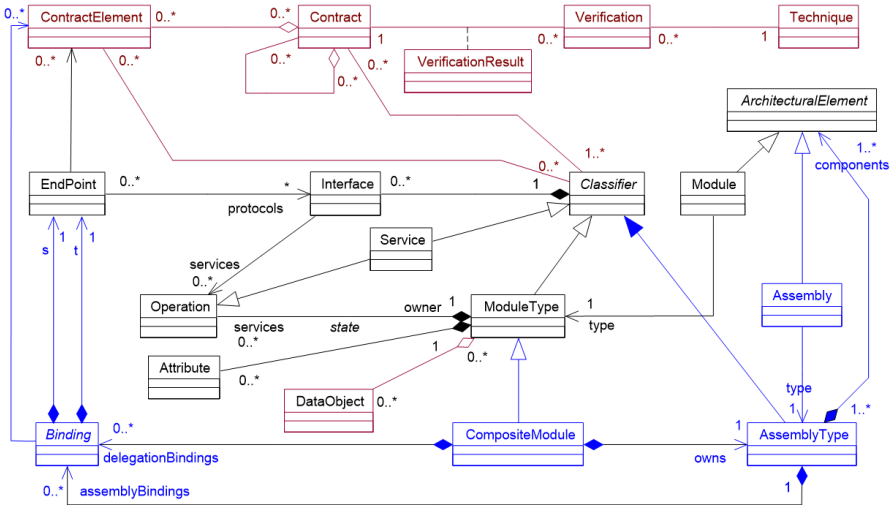

Fig. 3 Core RMS Configurations

ciple ④-abstraction). The *orchestration* of composite services is realised via the delegationBindings concepts. Contracts are associated to end points and interfaces and enable the principle ⑦ (early verification). This point is outside the scope, but the reader will find details in [2].

A **configuration** is basically an assembly of modules with initialisation information (the values for Attributes and Data objects). Note that a configuration can be "implemented" by configuration services to control the consistency of the input parameters.

4.2 RMS-Control

Modules as well as interfaces and services are interacting actors and have a behaviour defined by dynamic expressions (*cf.* Fig. 4). For example, the behaviour of an operation is defined by a finite state machine (FSM) where the transitions are labelled by (atomic) actions. Since FSM denotes widespread formalisms, details are omitted here. In this layer, it is assumed a service-based communication with message send and signals. No matter what the implementation of communications is, a service middleware enables to reason at the model level *e.g.* to verify contracts, according to principle ⑦, or to simulate the configurations (*e.g.* to compute KPIs).

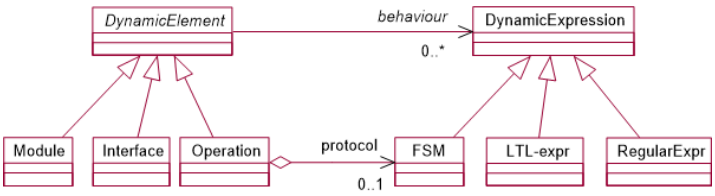

Fig. 4 RMS Dynamics

Several operations/services can run in parallel in a Module. The *choreography* is implicitly guided by operation and service calls but can be control by guarded actions (critical sections in concurrent processes). In CompositeModules, this parallelism is implicitly distributed on the component modules but can be controlled by *orchestrating* the delegationBindings through FSM protocols. Note that string encapsulation prevents component modules to collaborate with other modules than the other components modules of the same composite (assembly bindings) and the composite it-self (delegation bindings).

4.3 RMS-Production

The production layer includes every (abstract) concepts needed to organise the production process: products, orders, storage, etc. We assume here a Product layer that defines products, which is a common topic to all MESs. Modelling examples of orders, products, passive storage and containers are provided in [3]. Transportation

and active storage are supposed to belong to RMS modules. Scheduling heuristics are assumed to be found in the Intelligence layer.

The RMS reference model set no constraints on the MES control organisation. Basically, in a centralised version of MES (orchestration), an order manager schedules the orders (and recursive sub-orders), each order is a sequence of tasks (recursive sequences of sub-tasks until reaching atomic tasks). In a decentralised version of MES (choreography), an order manager is associated to every composite module to organise the production sub-process and enable partial reconfiguration or better runtime reconfiguration.

The MES dispatches orders on high-level modules according to product information (not explicit here). In [5], this coupling is designed by smart process/product decomposition of products, the processes are bound to resources and the products are bound to orders. In [12] orders are specialised in Product-Order and Resource-Order, each of them focusing on one side of product-order-resource collaboration. In [3], the Order aggregation pattern enables finer coordination with product and resource holons. For example, the tasks can be associated to resources only, while the orders would be associated to products only. Note that orders and tasks can also be independent of resources and products to perform management or information processing.

4.4 KPI Computation

In many MES, KPIs are handled in two standalone processes: the MES stores the production events in databases and the KPI application queries the databases to compute KPIs by filtering on event types and the timestamps events under the software developer's responsibility. Reconfiguration may influence both the KPI definition and computation and if they are considered as a whole, the entire KPI system is to be (costly) revisited. The idea in our reference model is to improve modularity (and reuse) (principle ②) while separating the Enterprise and the Production concerns (principle ⑥). The KPI definition KPI–MM is inspired from KPIML [18] that implements the ISO 22400 standard [14, 15]. It will always depend on the enterprise information system (*e.g.* ERP) and it is assumed here they focus on products (not the production resources). The main concerns are first how to define explicitly aggre-

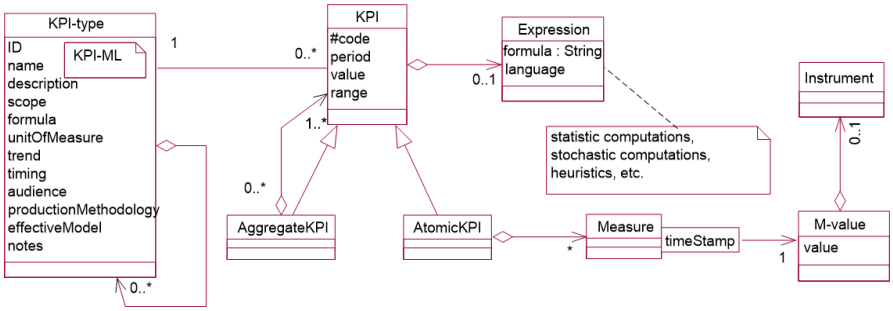

Fig. 5 RMS KPI

gated KPIs of the management dashboard from production KPI and values (left part of Fig. 5) and second to feed KPI computations with values in respect with time periods (right part of Fig. 5). KPI s are defined according to KPI–types (KPI definitions in KPIML) and the KPI computation tree is made explicit through aggregation, where the expressions use the KPI codes. The tree leaves are computed on measures (metrics) on the production system by the means of Instruments: (1) Counter: inc/dec/raz - value, (2) Timer: start/stop/restart/reset - value and (3) Sensor: trigger - time stamped events. These instruments are plugged on the RMS–Production over products, modules (and their resources), orders...

This section sketched the main concepts of the reference architecture. Next section will illustrate some of them.

5 Application on a Simplified Case Study

A schematic representation of the case study is presented in Fig. 6 using Flexsim. It consists in a small manufacturing system with 3 workstations. In workstation 1, denoted Module 2 in Fig. 6, a robot place two components on a pallet, itself placed on the conveyor. The pallet moves to workstation 2 (Module 3), where a human operator takes one of the component to be processed on the machine denoted Processor1. The pallet then moves to workstation 3 (Module 4), where the operator takes the second component to Processor2. Both operators bring back the processed components to the pallet, which then leaves the workstation to get to the exit, where a cart, denoted Transporter1, transports the components one by one to the exit buffer. The pallet is free again to run a next cycle. On the conveyor, several pallets can be used in parallel.

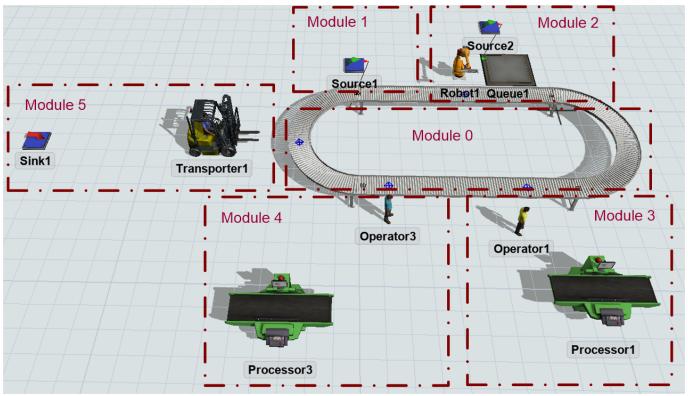

Fig. 6 Workshop

Model

The case study layout and module map are showed in Fig. 6. Every physical part of the system has to be represented somewhere in Fig. 7, a module model depicted with the UML component diagrams,

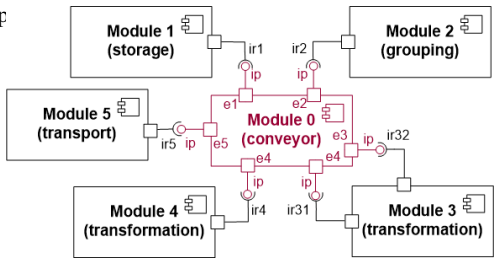

Fig. 7 Workshop modules model

where rectangles are modules, square are end-points, lollipops are provided interfaces and semi-circles denote required interfaces. In this case study, the modules do not interact directly; all are connected to the Module 0 (conveyor) that plays a role of communication bus. The end-point e_i (pallet blocker) is represented five times for readability reasons with the same provided interface ip [stopPallet , raiseUp, pick, put, rotate]. However, the corresponding required interfaces ir_i are different by default. Examples of high-level services are following: (1) Module 1 [addEmptyPalle], (2) Module 2 [addTwoProductsOnPallet], (3) Module 3&4 [takeProductFromPallet, processProduct, putProductOnPallet], (4) Module 5 [takeProductFromPallet, storeProduct]. Of course running these services require interactions with Module 0 services. The fluid and energy flows are not defined here, but they are considered as data and constraints in the interface and services. In Fig. 6, Modules 3 and 4 are clearly composite modules with two resources each, an operator and a processor. In the assembly of Fig. 7, only the module interfaces are available (black-box encapsulation). Note that Module 3 has two end points access with Module 0 because Operator1 takes a product in one point and puts the product that has been transformed by Processor 1 in another place of the conveyor (Module 0).

The RMS–Control layer adds dynamic behaviours to modules. For example, the left part of Fig. 8 shows the dynamic behaviour of service addTwoProductsOnPallet that loads two products from the store on the current pallet. The right part of Fig. 8 shows another possible behaviour that infinitely loads products on pallets.

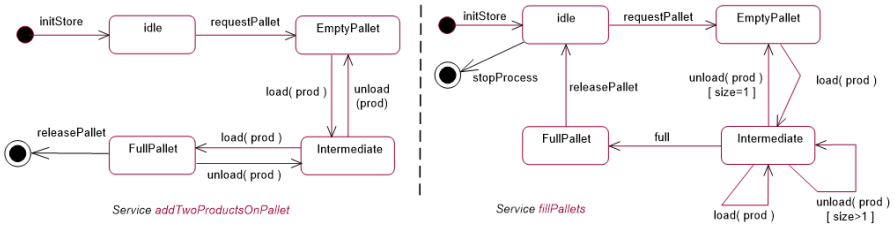

Fig. 8 Dynamic behaviour *addTwoProductsOnPallet*

The Products layer describes data objects: a single instance of ProductType that describes the operations to be performed on the Product instances that will travel between the modules. Each instance is identified by a product_id.

The KPI–MM includes the following performance indicators instances: (1) *OEE index = Availability * Performance rate * Finished goods ratio* ; (2) Finished goods ratio - The FRG is the ratio of the good quantity produced (GQ) to the consumed material (CM). *FRG = GQ / CM* (3) The finished goods inventory shall be the amount of acceptable quantity which can be delivered. *FGI finished goods inventory* (4) The quality ratio is the relationship between the good quantity (GQ) and the produced quantity (PQ). (5) Throughput rate = PQ/AOET (Actual order Execution Time) (6) Inventory turns *Inventory turns = TH / average inventory*.

The KPI Evaluation needs metrics and equipments to compute the KPIs. For each measure that enters in AtomicKPI expression, an instrumentation is provided. The OEE

index is instrumented by 3 timers for each module: waiting for product, waiting for pallet, production. The timers provide values for the availability and the performance. The AOET is implemented by one timer per order, starts when the production starts with the first product instance, stopped when the last product instance is manufactured. The consumed material (CM) value is instrumented by a counter associated to Module 5 that is incremented each time a product is stored in the inventory sink1 it is also the produced quantity (PQ). The good quantity (GQ) value is instrumented by a counter associated to Module 2 that is incremented each time a product is taken from the inventory source2. The number of processed products are stored in counters associated to modules 3, 4 and 5. The Inventory turns is implemented by a counter associated to Module 1 that is incremented each time a product is picked from the inventory source1. The counter and timers are time-stamped vectors when periodic rates are required (for time evolution feedback in simulation dashboards).

6 Discussion

This section points out relevant information on the applicability of the approach.

It is necessary to propose to the reconfiguration operator a dedicated language so that he can model the expected configuration in a comprehensive manner. At the same time, it is a major concern to avoid large-scale (complex) languages such as UML or SysML, because their semantics is a true weakness for complex systems. Our experience showed that using small dedicated languages (DSL) expose clear semantics for each modelled aspect. Hence Fig. 2 structures the reference architecture as a collection of small DSL, each of them being adapted to a different perspective of the configurations. However, defining the adequate glue is also crucial since it provides the consistency of the whole model. We suggest to this end the use of non-intrusive approaches for model composition to preserve the semantics of each language.

Section 4 presents a reference model, not yet an implementation. Currently, we plan an implementation for simulation purpose that should be later plugged to real systems according to a model-driven approach, where a generic framework previously defined provides the execution engine [9]. Many design issues are to be solved including production task management, data management and communication middlewares.

Having a collection of DSL to model manufacturing configurations is the cornerstone of the reference model. Next layer is to dispose of reconfiguration languages to handle the true reality of modifying or changing manufacturing systems in factories. For sake of space, we leave this point to future work, but the idea is to consider each configuration as a state in a finite state machine (or graph) where the edges support the cost effort of reconfigure. We suggest to process according to the same approach we chose for KPI, by aggregating local low-level costs to a vector cost at the top-level of the manufacturing systems.

7 Conclusion

Conceptual models are missing to capture the particularities of reconfigurable manufacturing systems (modularity, extensibility, substitutability, encapsulation...). In this paper we propose a reference model for RMS with the aim of being applicable by considering the different points of view on manufacturing systems. We detailed parts of these points of views by the means of packages and driving the models toward implementation. For example, the KPI vision is to be instrumented by a system observation instead of leaving this aspect to software providers. We illustrate on a simple case study how this can be combined to describe a manufacturing system.

We plan an implementation of this reference model for simulation purpose where the performance of each configuration is evaluated. This implementation should be later plugged to real systems according to a model-driven approach where a generic framework provides the execution engine [9]. Domain specific languages, dedicated to the points of views will be composed in order to preserve the consistency of each point of view (modelling simplicity, verification adequacy). Many design issues are to be solved including production task management, communication [4]. Such an implementation is the first and short-term perspective. In parallel, this reference model is the corner stone for reasoning on reconfigurations capabilities. Configurations are then considered as performance objects enriched by cost features and reconfigurations are operations on the objects; costs are also associated to the operations. Such a system would enable to evaluate the cost/profit balance of reconfiguration.

Acknowledgments

Authors thank financial support from the French National Research Agency (ANR) under the RODIC project, grant number ANR-21-CE10-0017.

References

1. Alvarez, K., Reyes, J.: Reconfigurable manufacturing system based on the holonic paradigm for the die-cutting process in a sports shoes company. In: M.V. García, F. Fernández-Peña, C. Gordón-Gallegos (eds.) Advances and Applications in Computer Science, Electronics and Industrial Engineering, pp. 19–36. Springer Singapore, Singapore (2021)
2. André, P., Cardin, O.: Trusted services for cyber manufacturing systems. In: T. Borangiu, D. Trentesaux, A. Thomas, O. Cardin (eds.) Proceedings of SOHOMA 2018, pp. 359–370. Springer IP (2018)
3. André, P., Cardin, O.: Aggregation patterns in holonic manufacturing systems. In: T. Borangiu, D. Trentesaux, P. Leitão, O. Cardin, L. Joblot (eds.) Proceedings of SOHOMA 2021, pp. 3–15. Springer International Publishing, Cham (2022)
4. André, P., Cardin, O., Azzi, F.: Multi-protocol communication tool for virtualized cyber manufacturing systems. In: T. Borangiu, D. Trentesaux, P. Leitão, O. Cardin, S. Lamouri (eds.) Proceedings of SOHOMA 2020, *Studies in Computational Intelligence*, vol. 952, pp. 385–397. Springer (2020)
5. Blanc, P., Demongodin, I., Castagna, P.: A holonic approach for manufacturing execution system design: An industrial application. Engineering Applications of Artificial Intelligence **21**(3), 315–330 (2008)

6. Bortolini, M., Galizia, F.G., Mora, C.: Reconfigurable manufacturing systems: Literature review and research trend. Journal of Manufacturing Systems **49**, 93–106 (2018)
7. Cerqueus, A., Delorme, X.: Evaluating the scalability of reconfigurable manufacturing systems at the design phase. International Journal of Production Research pp. 1–14 (2023), publisher: Taylor & Francis
8. Cerqueus, A., Gianessi, P., Lamy, D., Delorme, X.: Balancing and Configuration Planning of RMS to Minimize Energy Cost. In: B. Lalic, et al. (eds.) Advances in Production Management Systems. Towards Smart and Digital Manufacturing, IFIP Advances in Information and Communication Technology, pp. 518–526. Springer International Publishing, Cham (2020)
9. El Amin Tebib, M., André, P., Cardin, O.: A model driven approach for automated generation of service-oriented holonic manufacturing systems. In: T. Borangiu, D. Trentesaux, A. Thomas, S. Cavalieri (eds.) SOHOMA, pp. 183–196. Springer IP (2019)
10. Beauville dit Eynaud, A., Klement, N., Roucoules, L., Gibaru, O., Durville, L.: Uml based reconfiguration rate analysis of assembly line depending on robot integration. IFAC-PapersOnLine **51**(11), 1168–1173 (2018), iNCOM 2018
11. Fotsoh, E.C., Mebarki, N., Castagna, P., Berruet, P., Quintanilla, F.G.: Towards a reference model for configuration of reconfigurable manufacturing system (RMS). In: B. Lalic, et al. (eds.) Advances in Production Management Systems. IFIP WG 5.7, *IFIP Advances in Information and Communication Technology*, vol. 591, pp. 391–398. Springer (2020)
12. Gamboa Quintanilla, F., Cardin, O., L'anton, A., Castagna, P.: A modeling framework for manufacturing services in service-oriented holonic manufacturing systems. Engineering Applications of Artificial Intelligence **55**, 26–36 (2016)
13. Guan, X., Dai, X., Qiu, B., Li, J.: A revised electromagnetism-like mechanism for layout design of reconfigurable manufacturing system. Computers & Industrial Engineering **63**(1), 98–108 (2012)
14. ISO: Iso22400: 1—automation systems and integration—key performance indicators (kpis) for manufacturing operations management—part. 1: Overview, concepts and terminology (2014)
15. ISO: Iso22400: 2—automation systems and integration—key performance indicators (kpis) for manufacturing operations management—part. 2: Definitions and descriptions (2014)
16. Kaiser, J., McFarlane, D., Hawkridge, G., André, P., Leitão, P.: A review of reference architectures for digital manufacturing: Classification, applicability and open issues. Computers in Industry **149**, 103,923 (2023)
17. Morgan, J., Halton, M., Qiao, Y., Breslin, J.G.: Industry 4.0 smart reconfigurable manufacturing machines. Journal of Manufacturing Systems **59**, 481–506 (2021)
18. Muhammad, U., Ferrer, B.R., Mohammed, W.M., Lastra, J.L.M.: An approach for implementing key performance indicators of a discrete manufacturing simulator based on the iso 22400 standard. In: 2018 IEEE Industrial Cyber-Physical Systems (ICPS), pp. 629–636 (2018)
19. Petroodi, S.E.H., Eynaud, A.B.D., Klement, N., Tavakkoli-Moghaddam, R.: Simulation-based optimization approach with scenario-based product sequence in a reconfigurable manufacturing system (rms): A case study. IFAC-PapersOnLine **52**(13), 2638–2643 (2019), 9th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2019
20. Rösiö, C., Aslam, T., Srikanth, K.B., Shetty, S.: Towards an assessment criterion of reconfigurable manufacturing systems within the automotive industry. Procedia Manufacturing **28**, 76–82 (2019), 7th International conference CARV2018
21. Sabioni, R.C., Daaboul, J., Le Duigou, J.: An integrated approach to optimize the configuration of mass-customized products and reconfigurable manufacturing systems. The International Journal of Advanced Manufacturing Technology **115**(1), 141–163 (2021)
22. Singh, P.P., Madan, J., Singh, H.: Composite performance metric for product flow configuration selection of reconfigurable manufacturing system (rms). International Journal of Production Research **59**(13), 3996–4016 (2021)