Multi-Robot based Control System

Atef Gharbi[1]
Faculty of Computing and Information Technology[1]
Northern Border University
Rafha, KSA
Institut National des Sciences Appliquées et de Technologie (INSAT), LISI[1]
Université de Carthage
Tunisia

***Abstract*—One of the most important challenge in Robotic Flexible Manufacturing Systems (RFMS) is how to develop a Multi-Robot based control system in which the robot is able to take intelligent decision to a changing environment. The problematic is how to ensure the flexibility with the proposed multi-robot based control system based on triggering strategies. The flexibility of the whole system is expanded by the capacity of the flexible robots to effectively ensure tasks assigned to it. Through this paper, three contributions can be presented: (i) the RFMS based Control Architecture by presenting in details the main components and methods, (ii) the planning model, and (iii) the different levels of flexibility in RFMS.**

Keywords*—Robotic Flexible Manufacturing Systems (RFMS); multi-robot based control system; RFMS control architecture; planning model; flexibility

I. Introduction

Nowadays, Flexible Manufacturing Systems (FMS) are facing widely frequent market changes determined by world-wide competition, new customers' requirements, continuous evolution of software and hardware, and the rapid introduction of new products [1]. Flexible Manufacturing Systems must ensure high quality products at acceptable costs and react rapidly to new market and products changes [2]. FMS can meet product changes, but they cannot respond to structural changes. In fact, the manufacturing systems are not able to face the dynamic changing environment due to their static control structure [3-5]. To react rapidly to the quickly changing environment, Robotic Flexible Manufacturing Systems (RFMS) based on multi-robot control system is considered as a good solution having the properties such as adaptability and flexibility [6]. Multi-robot based control system is an emerging solution which is becoming more and more popular as it helps to decentralize the decision in the control system [7]. Nowadays, there is a huge number of research activities that has been approved in this sense [8-10]. It is basic for Robotic Flexible Manufacturing Systems to have capacities such as autonomy, flexibility and adaptability. The RFMS framework based on Multi-Robot based control is intended to meet these criteria. A flexible manufacturing system can be applied either on static or dynamic system [11]. Robotic Flexible Manufacturing Systems can be used in many fields such as: medicine [12, 13], thermodynamic domain [14], optimal numerisation [15], motion [16], assembly system [17], automotive [18], fuzzy system [19].

The Robotic Flexible Manufacturing Systems can be based on either customisation or design. Robotic Flexible Manufacturing Systems based on customisation means satisfy the customers' needs during the design process of manufacturing systems leading to more time and effort before completing it [20].

Robotic Flexible Manufacturing Systems based on design means the ability of the change to obtain new robotic manufacturing systems based on existing ones as required, simply and economically [21]. The flexibility in design permits generating new manufacturing systems effortlessly by ensuring the required modifications from the existing ones, and the development cost can be significantly decreased [22].

In this paper, the architecture as well as the behaviour of intelligent flexible robots are presented. Therefore, the contributions are based on the following operations: (1) Firstly, design of flexible software architecture especially for a flexible robot. (2) Secondly, specification of the planning model ensured by the flexible robot. (3) Thirdly, the different levels of flexibility in Robotic Flexible Manufacturing System. To approve these contributions, the proposed methodology is applied to a benchmarking system.

Step 1: Define the high-level architecture of the RFMS.

On the basis of the control and flexibility objectives, the multi-robot control system is designed till the single controlled device and the related automation tasks. It is important for the system to be designed in a way ensuring capabilities such as flexibility. To do so, the multi-robot control system is conceived to incorporate several self-flexible levels to respond quickly to any changes occurring in the environment.

Step 2: Define the planning ability.

In the Multi-Robot Based Control System, the planning ability is considered as a very important point to study that's why it is defined how it is implemented. In fact, a plan is considered as a state-transition model.

Step 3: Define the flexibility ability.

In the Multi-Robot Based Control System, a Flexible Robot is defined what means. After that, a study on how the Flexible Robot ensures the flexibility. In order to cover a wide range of the production policies, the flexible robot must ensure several flexibility levels that can be categorized in the following ways: the product family, the product variant, the plan, the task, the

skill, the failure, the production control, the adaptation and the configuration. These different levels of flexibility will be detailed later in the paper.

The remainder of this paper is organized as follows: Section 2 introduces the state of art. Section 3 presents the production system benchmark used as running example. Section 4 describes the Multi-Robot based Control Architecture. Section 5 defines the planning model. The Section 6 presents the different levels of flexibility in RFMS. Finally, the conclusion and future work are summarized in the last section.

II. State of the Art

To define well a robot control system, a special attention is given to its architecture. A huge number of research papers was presented to define it. The first classification is based on Knowledge Utilisation based on the way the robot uses its knowledge to perform action. In this first classification, there are Competitive approach and Collective approach. The competitive approach is based on the use of a single criterion to take decision [23]. The Collective approach enables to take in consideration many criteria to make decision [24]. The competitive approach can be categorized into five types, namely lookup-based [25], finite state machine [26-27], priority-based or hierarchical-based [28], goal-based [29] and utility-based competitive approaches [30].

The second classification is based on Knowledge Design. The intelligent robot can be represented through a defined architecture that can be deliberative, reactive or hybrid. The deliberative architecture (called also hierarchal architecture, top-down, knowledge-based approach, or explicit-based approach) is the most used in the artificial intelligence [31]. The deliberative architecture consists of vertical layers where each layer is based on the data sent by the previous one. In general, a robot senses the environment, plans and executes to achieve a goal.

The reactive architecture (called also bottom-up, behaviour-based architecture or implicit-based approach) is based on a mapping between perception (provided by sensors) and action. The reactive architecture is considered as horizontal architecture where the different behaviors can be executed in parallel [32].

The hybrid architecture is more commonly used especially to control robot as the reactive aspect permits to take action in real-time to the perception of the dynamic environment and the deliberative aspect enables to plan future actions to satisfy a goal [33].

Each architecture has its own strengths and weaknesses. The deliberative architecture is likely to have higher computational cost than the reactive approach due to data sent between vertical layers. In addition, the deliberative approach is more complicated due to a complete knowledge has to be provided. However, the reactive architecture is less flexible than the deliberative one because behaviors cannot be modified as much as in the deliberative architecture (although it is considered easy to implement). Therefore, purely deliberative and reactive architecture are not considered suitable for a complex system. In this context, the hybrid architecture gets strengths as well as weaknesses of the two approaches, thus why we focus on how to balance the two approaches in this paper.

III. Production System Benchmark

As much as possible, the contribution will be illustrated with a simple current example called RARM [34]. It is described informally, but it will be used as an example of the various formalisms presented in this article. The production system benchmark RARM is depicted in Fig. 1.

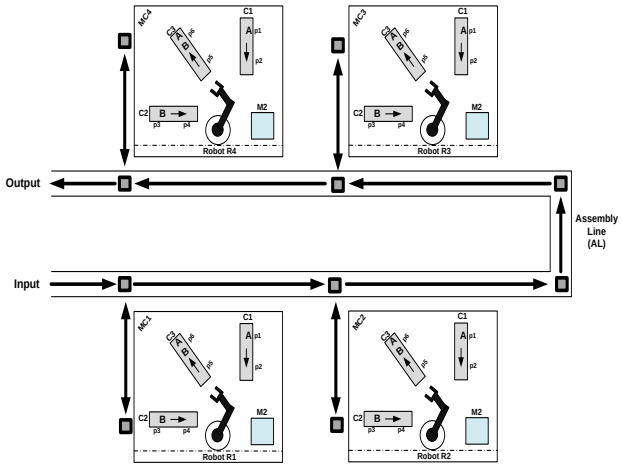

Fig. 1. The Production System Benchmark RARM.

The whole manufacturing system is divided into two main parts (Fig. 1): Assembly Line (AL) and Manufacturing Cell (MC). The Manufacturing Cell named as MC1, MC2, MC3 and MC4 are connected through the Assembly Line (AL) to enable the flexible robots moving from one MC to another.

Each Working Place contains three conveyors (C1, C2 and C3), a processing–assembling unit (machine M), a flexible robot R and additional sensors. Workpieces to be treated as they come irregularly one by one. The workpieces of Type A are carried via the conveyor C1, and the workpieces of Type B, via the conveyor C2. Only one workpiece can be on the input conveyor. The flexible robot is used to load and unload workpieces between the processing–assembling unit and the storage equipment (Input/Output). Firstly, the flexible robot carries a workpiece from the Input storage to the processing–assembling unit, which is processed by the machine M. After processing, the flexible robot transports the finished workpiece to the Output storage.

IV. High-Level Robot-based Control Architecture

The traditional methodology in designing robots has been to design the hardware and the software according to what it should do. Traditional robots can execute specific tasks, but they are not flexible, and therefore applications assigned to them depend on their physical structure and their controller abilities. Creating Flexible Robotic Manufacturing System is facing hardware and software challenges.

While existing survey papers on Flexible Robotic Manufacturing System have studied the architecture and hardware feature of robots, in this paper, a special attention is

given to the challenging issues emerged when developing flexible robots. Thus, the main problem to resolve is arising when the flexible robot perform tasks through some flexibility abilities.

To react rapidly to the quickly changing environment, it is basic for the framework to have such capacities as adaptability and flexibility. The Multi-Robot based control framework is designed to meet these needs. To ease the system flexibility, robots would be itself flexible. To do so, every Flexible Robot belonging to the Multi-Robot system has its own Goal to achieve and can generate a plan associated to this goal. This policy helps the Flexible Robot to select an appropriate plan of tasks to be executed.

In Fig. 2, a Flexible Manufacturing System Meta-Model is presented. Each Product is assumed to have its own Family. A Product_Variant is considered as a specific case of Product_Family. To ensure a Product_Variant, a list of Plan has to be executed and is composed of Task_Manager. Each Task_Manager has some inputs which are Events and uses some Resources, perceive data through Receptors and execute commands by Effectors. The flexible robot needs to have some specific Skill_Manager to execute well a task.

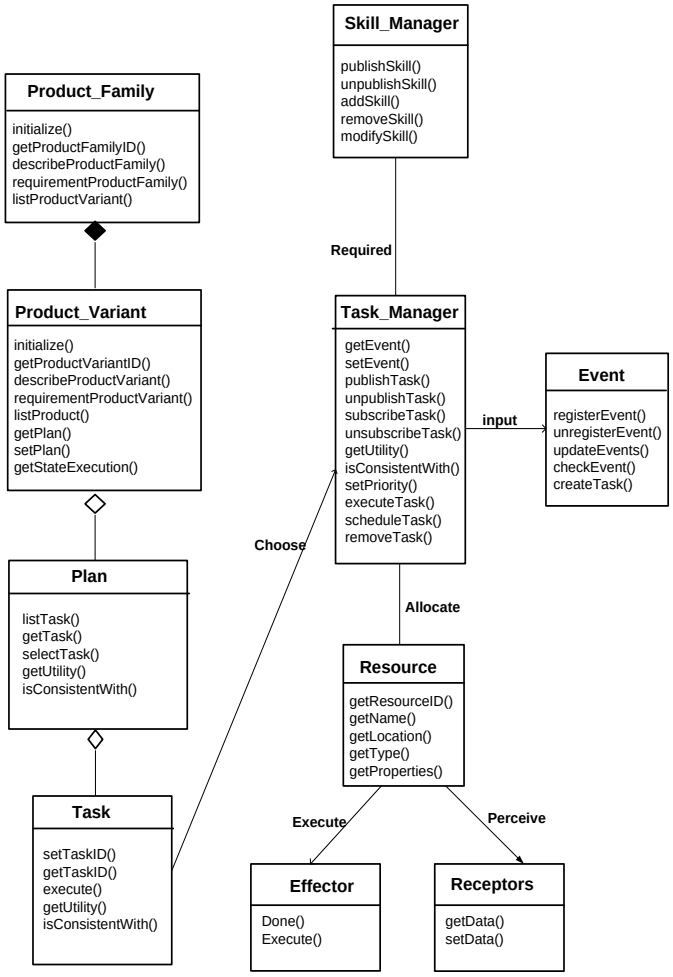

Fig. 2. Conceptual Meta-Model for a Flexible Manufacturing System.

Where

- Product_Family: It is a set of similar products having common tasks. In the production system RARM, it is possible to produce two potential product families simultaneously. To produce these product families, three types of machines need to be installed: Drill, Load, and Assembly. All of these machine types have a modular structure that allows adding/removing services. Based on these services, each machine type can have different configurations with different abilities and/or skills. Therefore, the flexible robot chooses the right machine configuration and decides the best production policy based on the machine availability and their cost structures.

Running example: In the benchmarking production system, two types of product families are defined: product family type and product family type. It is possible to process the two potential product families simultaneously. The product family type is treated firstly by RMC1 and then by RMC2. The product family type is handled firstly by RMC2 and then by RMC3. As for RMC1 (resp. RMC2, RMC3), it consists of a drilling machine (resp. milling machine, assembling machine). The buffer can store the finished workpieces and workpieces waiting to be processed.

- Product_Variant: is the same product but having different size, color, materials.

Running example: For the product family type , there are three possible product variant (i) the first production variant consists of inserting an A-work piece (through the conveyor C1) into the processing center M to be treated, then it is evacuated by the robot to the output conveyor C3; (ii) the second production variant consists of inserting a B-work piece (through the conveyor C2) into the processing center M to be treated, then it is evacuated by the robot to the output conveyor C3; (iii) the third production variant consists of inserting an A-work piece into the processing center M to be treated, then a B-work piece is added in the center and the two work pieces are finally assembled.

- Plan: The Flexible robot may have many tasks which are inconsistent. Therefore, the plan is composed of consistent tasks. The plan is composed of a set of tasks for a given product variant that can be either fixed or variable. The aim is to regroup as much as possible of tasks to be included in the same plan. Thus, the ***selectTask*** method is used to add a task in the Plan and the ***getTask*** method permits to return all the tasks related to the same Plan. Similar to Task, the ***isConsistentWith*** method is used to verify the consistency between two plans and the ***getUtility*** method enables to choose a plan among many concurrent existing Plans. If a plan has been chosen, the ***commitGoal*** method is used to create a new Goal instance.

- Task: In general the task objects are associated to the event object. There is a relation one-to-many between task and event. The methods ***setEvent*** and ***getEvent*** are used to set a link between task and associated events. To check the consistency between two tasks, the

isConsistentWith method is used. The Flexible robot has to choose between several tasks existing at the same time (which some of them can be inconsistent) based on the task's utility through the ***getUtility*** method. In some circumstance, the Flexible robot decides to select one or several consistent tasks to constitute a Plan with the use of ***selectAsPlan*** method. For each task, all exceptions are enumerated and trigger events are determined.

Running example: The set of actions is {Ci1_left, Ci1_right, Ri1_left, Ri1_right, Ci2_left, Ci2_right, Ri2_left, Ri2_right, Ci3_left, Ci3_right, Ri3_left, Ri3_right, takei1, takei2, takei3, loadi1, loadi2, loadi3, puti1, puti2, puti3, processi1, processi2}

Where:

- Ci1_left (resp. Ci1_right) means a workpiece of type A is moved to the left of conveyor Ci1 from position p1 (resp. p2) to position p2 (resp. p1).
- Ri1_left (resp. Ri1_right) means the Robot ri taking a workpiece of type A is moving to the left (resp. to the right) from the position p2 of conveyor Ci1 (resp. the processing unit Mi) to the processing unit Mi (resp. the position} p2 of conveyor Ci1).
- takei1 (resp. takei2, takei3) means the Robot ri is currently taking a workpiece of type A (resp. B , AB).
- loadi1 (resp. loadi2, loadi3) meansthe fact of loading a workpiece of type A (resp. B , AB).
- puti1 (resp. puti2, puti3) means the Robot ri is currently putting the workpiece of type A (resp. B , AB).
- processi1 (resp. processi2) means the fact of processing a workpiece of type A (resp. B).
- Event: the Flexible robot can update its knowledge about the environment through sensors. The Flexible robot can register to a specific event (this is done by the ***registerEvent*** method) or unregister (through the ***unregisterEvent*** method). The events associated to the robot are considered independent. Whenever the Flexible robot receives a new event, it checks firstly if there is a need to create a new task (this is done by the ***checkEvent*** method). If the condition is satisfied, a new task is created (this is ensured by the ***createTask*** method). All tasks arise from the Flexible robot's perception.
- Resource: Each manufacturing system is composed of a set of resources (e.g., machines, tools, grippers, conveyors, transport devices, etc.). Each resource performs a distinct function. It is possible to find a pool of more than one resource that has the same function.
- Receptor: the flexible robot knows its environment through sensors. Thus, the data provided by the sensors present the robot's vision of its environment. The perception parameters have to be defined and the robot must know how to interpret the data.

Running example

- The sensor sens1 (respectively sens2) is used to verify if there is a workpiece at the position p1 (respectively the position p2) on the conveyor C1;
- The sensor sens3 (respectively sens4) checks for the existence of a workpiece at the position p3 (respectively the position p4) on the conveyor C2.
- Effector: the flexible robot can execute the task using the effector. For each effector, a behavior is proposed to judge the requests to it.

Running example

- The effector act1 (respectively act2, act3) ensures the movement of the conveyor C1 (respectively C2, C3);
- The effector act4 rotates a robotic agent;
- The effector act5 elevates the robotic agent arm vertically.

V. PLANNING MODEL

The planning model means how the flexible robot should act to decompose the problems into subproblems to obtain the whole solution that the flexible robot must apply. The planning model of the flexible robot leads to a very huge number of possibilities which the flexible robot will have to take in consideration again in order to retain only the valid possibilities that should be kept.

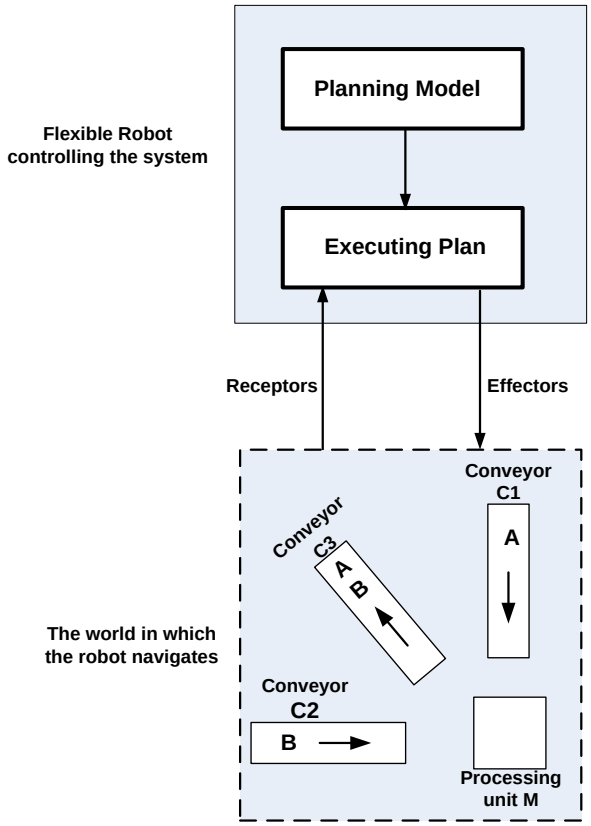

Fig. 3. The Conceptual Planning Model.

The planning model is based on two necessary elements: the receptors (i.e. a set of sensors to get data about the external environment in which the flexible robot is existing) and the effectors (i.e. a set of actuators to realize the flexible robot's tasks) [35]. Fig. 3 shows a conceptual model of the flexible robot including two components: the planning model, and the executing plan [36, 37]).

To be more specific, the planning model is based on state-transition model where Σ representing the world is a finite state-transition system, i.e., a triple $\Sigma = (S; A; \gamma)$, where S is a finite set of states, A is a finite set of actions, $\gamma : S \times A \rightarrow 2^S$ is a state-transition function. If $(s, a) = \varnothing$; then it is said that a is not applicable to s or not executable in s.

Given a state transition system Σ , the aim of planning is to determine which actions to execute to which states in order to realize some objectives when starting from a given situation. A plan is a solution that determines the appropriate actions to reach the goal. The objective can be specified by a goal state s_g or a set of goal states Sg. The objective can be obtained by any sequence of state transitions that ends at one of the goal states. The planning model necessitates the descriptions of , the initial state before applying the plan, and the desired objectives (e.g., to reach a set of states that satisfies a given goal condition). Therefore, the planning model's objective is to produce a plan (i.e., an ordered finite sequence of actions) that puts Σ into any one of some finite set of states Sg.

More formally, the plan π is any sequence of actions $\pi = (a1 , \dots , ak)$, where $k \geq 0$.

The length of the plan is $| \pi | = k$ is equal to the the number of actions. If $\pi 1 = (a1 , \dots , ak)$ and $\pi 2 = (a'1, \dots , a'j)$ are plans, then their concatenation is the plan $\pi 1. \pi 2 = (a1 , \dots , ak, a'1, \dots ,a'j)$. The state produced by applying π to a state s is the state that is produced by applying the actions of π in the order given.

The plan π is executable in a state s0 if there is a sequence of states (s0; s1; … ; sn) such that for i = 1; … ; n,

$s_i = \gamma (s_{i-1} , a_2)$. In this case it is said that (s0; s1; … ; sn) is π's execution trace from s0, and $\gamma (s_0 , \pi) = s_n$ is defined. If sn satisfies the goal g, then it is said that π is a solution for the planning problem P = (O; s0; g).

The quality of a plan is measured by length, where the shorter of two plans is better under the same satisfaction degrees.

An action or a plan posts a set of goals G = {g1; g2; ...; gn}. This invokes the following process:

1) Loop over each goal gi:

a) Determine the set of plans filling the goal gi.

b) Keep only the plans which pre-conditions are satisfied.

c) For each remaining plan, check its mandatory resources.

2) Order the different goals gi according to its priority.

3) For every goal gi in order of priority.

a) If only one plan P realizes gi

then apply P

else // several plans

b) order each plan achieving the goal gi based on the length of the plan and how many resources it uses.

c) Choose the plan having the highest scoring.

Running example

Giving (S, A, Gs) where S = {si , i=1...n } is a set of states, A= {Ci1_left, Ci1_right, Ri1_left, Ri1_right, Ci2_left, Ci2_right, Ri2_left, Ri2_right, Ci3_left, Ci3_right, Ri3_left, Ri3_right, takei1, takei2, takei3, loadi1, loadi2, loadi3, puti1, puti2, puti3, processi1, processi2 i=1...n } is a set of actions, and Gs is the problem goal.

If the goal g = {workpiece in the processing unit} and the robot is at the initial position s1. Let:

- $\pi 0$ = (Ci1_left, takei1).
- $\pi 1$ = (load_i1, put_i1, process_i1, Ci1_right).
- $\pi 2$ = (Ci1_left, take_i1, load_i1, put_i1, process_i1, Ci1_right).

Then

$\pi 0$ is not considered as a solution because the final state is not a goal state;

$\pi 1$ is not a solution because it is not applicable to s1;

$\pi 3$ is the only solution because it is applicable to s1 and the final state is a goal state.

VI. How does the RFMS Ensure Flexibility?

The flexible robot controls the system through an event-triggering policy which means whenever an event related to the system (for example a resource failure), the flexible robot decides to take the right decision. In order to cover a wide range of the production policies, the flexible robot must ensure several flexibility levels that can be categorized in the following ways:

- Product Family flexibility means that the flexible robot is able to change-over production families to produce a new Product family which is feasible in terms of requirements (that means manufacturing facility procedure to ensure the production of each product family). In fact, Flexible Manufacturing Systems are designed to achieve several operations on many products grouped in families according to their operational requirements.

As it is illustrated in Fig. 4, each Flexible Robot has some Product to achieve. To do so, it determines the appropriate production plan and the tasks that can be executed. The internal behaviour of the Robot is defined as follow: firstly, the Flexible Robot evaluates the feasibility of the new product. If it is not feasible, then the Flexible Robot generates error, else it determines the list of tasks to be executed. Each task needs

some resources. If the Flexible Robot fails in achieving this task due to some missing resources, it can ask help from other robots able to provide the requested resources which are considered as Helping Robots.

Running example the flexible robot can switch from the product family type α to the product family type β.

In [38], the authors present a methodology to group products into families depending on similarities through a modified Jaccard similarity index.

$$S_{ij} = \frac{\sum_{m \in i} \sum_{n \in j} S_{mn}}{N_i N_j} \quad (1)$$

Where

i,j families,

m (resp. n) products of family i (respectively j)

Sij degree of similarity between families i and j

Smn degree of similarity between products m and n

Ni (resp. Nj) number of products in the family i (resp. j)

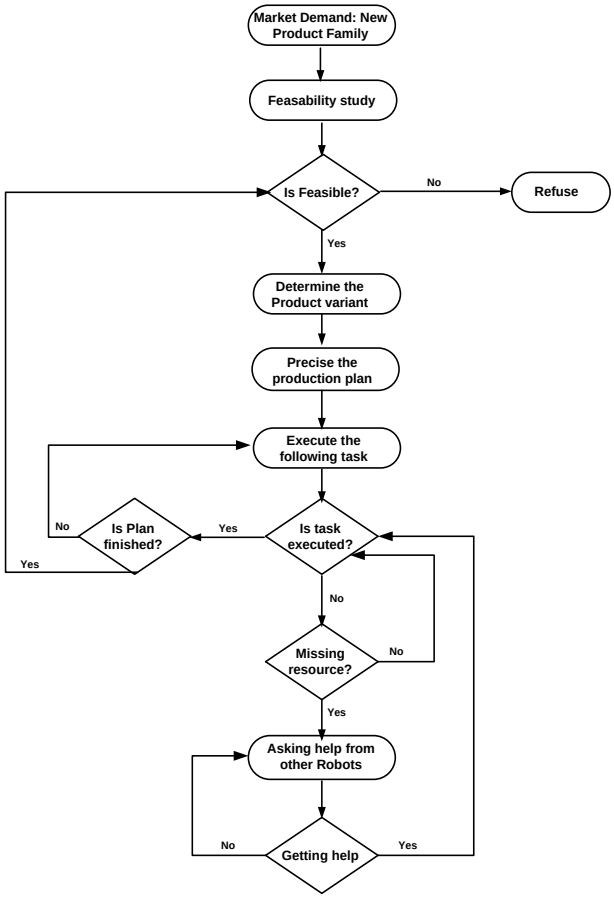

Fig. 4. Flexible Robot Behavior.

- Product Variant flexibility means that the flexible robot is capable to define the different possible configurations for the same product family (i.e. alternative configurations for each product family).

Running example the flexible robot can switch from the first production variant to the second product variant in case of shortage of workpiece A.

To measure the similarity degree between two products m and n, the Jaccard similarity coefficient Smn [38] is used which is defined.

$$S_{mn} = \frac{a}{a+b+c}, 0 \le S_{mn} \le 1 \quad (2)$$

Where a represents the number of common machines used to produce both the products m and n; b defines the number of machines used to produce the product m; and c represents the number of machines used to produce the product n.

- Plan flexibility means that the flexible robot is able to define the order of execution of the tasks to ensure the same plan.

Running example the flexible robot can switch from the following plan {C2 left, take2, load2, process2} to a new plan {load1, put1, process1, C1 right}. The following algorithm obtains a valid plan constituted by a task sequence (T1, .., Ti, Tj, .., Tn) where Ti.post-condition = Tj.pre-condition.

Algorithm Graph_generation()

Input: Node t(acti, pre-condi, post-condi), Precedence Graph G = (T, R)

Output: Precedence Graph G = (T, R)

Add t into T

For j in 1 to length(T) do

If (post-condj = post-condi)

Add r = (actj , acti) into R

End if

End for

- Task flexibility means that the flexible robot has the ability to manage the task switching with minimal effort.

A Task Precedence Graph G = (T, R) is used to provide a simple visual representation of a complex system by modelling the interactions and precedence relations among the different tasks where T is the set of tasks and R is the relationship between Tasks (precedence order).

Fig. 5 shows Task Precedence Graph composed of night Tasks (from T1 to T9), and illustrates how is the final production system configuration.

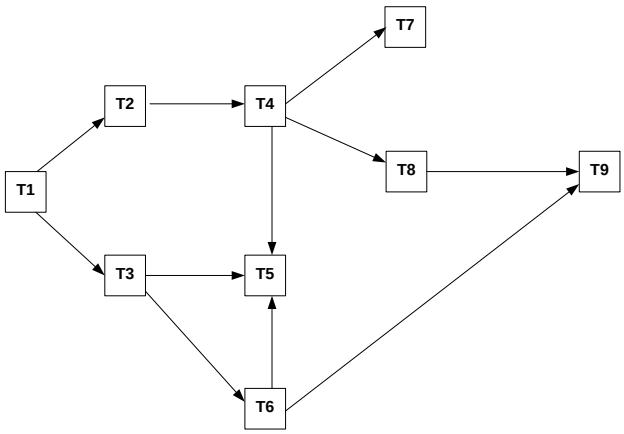

Fig. 5. Task Precedence Graph.

The list of tasks that can be executed is determined through the following algorithm.

Algorithm Task_Choice()

Input: Precedence Graph G = (T, R)

// T is the whole tasks

// R is the relationship between Tasks (precedence order)

Output: set of tasks can be executed by end actuators

Repeat

For each t in T do

If (indegree(t)=0) & (t. actuator = available) then

Return t

End if

End for

If (t.state = executed) then

t.actuator ← available

T ← T – {t} //Remove t from the whole tasks T

R ← R – t.outgoingEdge //remove its outgoing edges from R

End if

Until all tasks are executed

- Skill flexibility means that the flexible robot has the ability to change its different skills i.e. adding new skills, removing others, and modifying of several services composition, e.g., redesigning the services by adding a new one and eliminate others to be more flexible with the environment evolution.

Running example the flexible robot has the moving skill (forward/backward), it is possible to add on it the new skill turning (left/right).

- Failure flexibility describes the aptitude of flexible robot to deal with breakdowns and consequently guaranteeing continuation of production.

Running Example. The flexible Robot controlling the production system RARM consider many scenario in case of faults happen to physical components such as actuators, conveyors or machines.

- The first scenario involves a single conveyor C1 that transports A-work pieces to be processed by the machine unit.
- The second scenario involves a single conveyor C2 that transports B-work pieces to be processed by the machine unit.
- The third scenario involves two conveyors C1 and C2 that transports A and B-work pieces to be processed by the machine unit.

If the conveyor C1 is broken in the RARM Production System, then the flexible Robot has to apply the second scenario. If the conveyor C2 is broken in the RARM Production System, then the flexible Robot has to follow the first scenario. If the conveyor C1 and C2 are functioning well, then the flexible robot can apply the third scenario (Fig. 6).

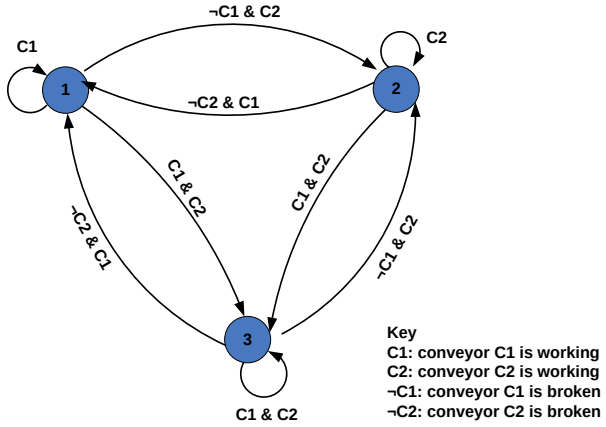

Fig. 6. Flexible Robot Behaviour in Case of Failure.

- Adaptation flexibility: Fig. 7(a), represents the normal case where there are: (i) two flexible robots, each one is existing in a station, (ii) Robot repository to help other robot facing problems and (iii) Gripper repository which permit robot to accommodate the product family (depending on the form and the geometry of the workpiece to be handled). This normal case is considered as the starting point upon which all adaptation scenarios are based.

In Fig. 7(b), rather than stopping the manufacturing system to repair the broken robot, the robot facing a problem can be substituted by another one. In Fig. 7(c), if the robot is broken and there is no other available robot, then the workpiece can be transferred to the second workstation to be processed.

In Fig. 7(d), a new suitable gripper is used to be convenient with the form of workpiece. The use of new grippers for the transfer of the workpiece facilitates the adaptation flexibility.

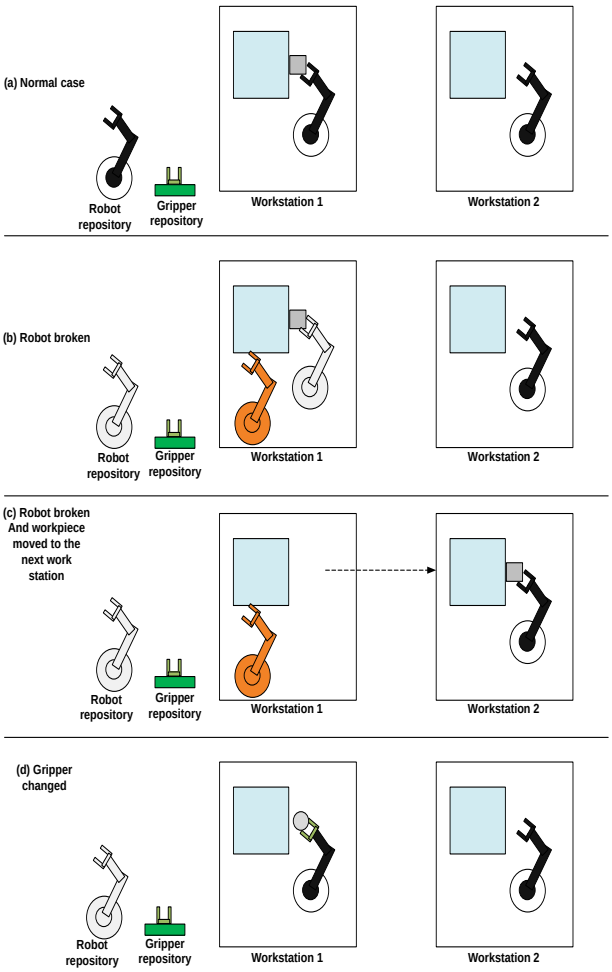

Fig. 7. Adaptation Scenarios.

- Configuration flexibility: Fig. 8 illustrates a decision graph representing all the set of possible configurations that can be executed by the flexible robot. It comprises six levels: (i) the first level represents the different product families (for example here, there are only two); (ii) the second level defines whether the Product Family is possible or not, (for each product family, there are two alternatives Possible or Not Possible); (iii) the third level specifies the different product variants that can be executed related to a specific Product Family if it is possible of course; (iv) the fourth level defines the availability to execute a product variant (for each product variant, the input indicates if is available or not); (v) the fifth level represents the different plans that can be executed for each product variant if it is available; (vi) the sixth level represents the different tasks to be executed for each plan. Each node of the tree graph is a decision point.

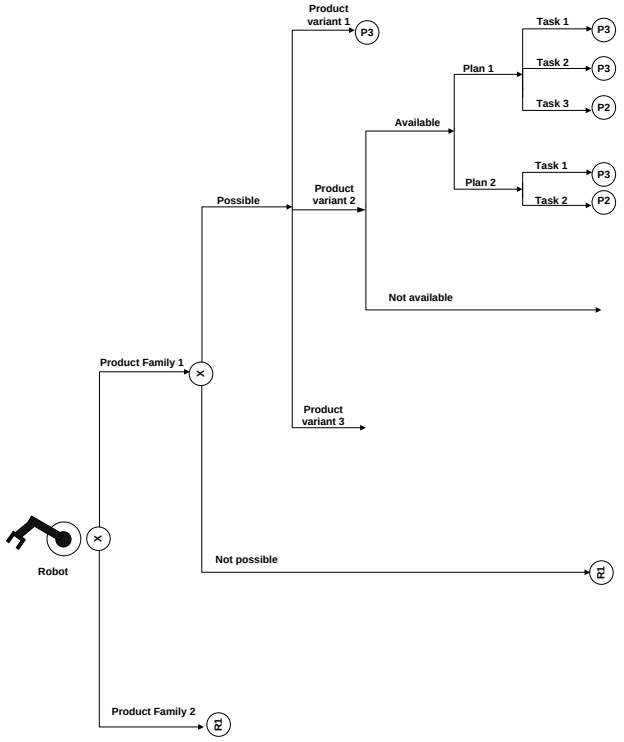

Fig. 8. Intelligent Robot Control Decision Making..

The total number of alternative solutions for each intelligent robot can be represented as:

$$R = \prod_{i} \prod_{s} \prod_{j} \prod_{a} \prod_{p} \sum_{t} R_{i,s,j,a,p,t} \quad (3)$$

Where

- R is the total number of possible configurations for the intelligent robot,
- i, is the product family index,
- s, Boolean parameter to indicate if the product family is possible (true or false),
- j, is the product variant index,
- a, Boolean parameter to indicate if the product variant is available (true or false),
- p, is the product variant plan,
- t, is the task index.
- Ri,s,j,a,p,t, is the complete configuration that the intelligent robot can choose. Based on the above analysis for each Product family, Product variant, Plan and the identification of suitable tasks to be executed in order.

VII. Conclusion

Robotic Flexible Manufacturing Systems (RFMS) is a suitable solution to accommodate changes and meet customers' needs such as autonomous decision, control, and flexibility to react rapidly to the quickly changing environment.

Through this paper, we consider the challenge how to implement RFMS, the proposed approach presents the following contributions: (i) firstly, the general approach is designed to define the basic architecture of RFMS by presenting in details the main components and methods, (ii) secondly, the control robot is defined how to deal with the planning, (iii) thirdly, the different manners in which the flexible robot can adapt the system are proposed. The robots behave more like they are thinking, by making a decision about action selection and predicting the effects of actions. Therefore, the problem is divided into three parts: the robot-based architecture, the planning model and the flexible robot behaviour.

The future work will be as the following. The methodology can be expended to include human-computer interaction. The Multi-Robot based control system can be ameliorated to allow robots to participate in multiple collaboration at the same time.

Acknowledgment

The authors gratefully acknowledge the approval and the support of this research study by the grant no -7436-CIT-2017-1-8-F- from the Deanship of Scientific Research at Northern Border University, Arar, K.S.A.

References

[1] Silva, A.; Ribeiro, R.; Teixeira, M. Modeling and control of flexible context-dependent manufacturing systems, Information Sciences, Volume 421, 2017, Pages 1-14, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2017.08.084.

[2] Liu, H.; Wu, W.; Su, H.; Zhang, Z. Design of optimal Petri-net controllers for a class of flexible manufacturing systems with key resources, Information Sciences, Volume 363, 2016, Pages 221-234, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2015.11.021.

[3] Gao, G.; Wang, J.; Yue, W.; Ou, W. Structural-vulnerability assessment of reconfigurable manufacturing system based on universal generating function, Reliability Engineering & System Safety, Volume 203, 2020, 107101, ISSN 0951-8320, https://doi.org/10.1016/j.ress.2020.107101.

[4] Zhang, Y.; Zhao, M.; Zhang, Y.; Pan, R.; Cai, J. Dynamic and steady-state performance analysis for multi-state repairable reconfigurable manufacturing systems with buffers, European Journal of Operational Research, Volume 283, Issue 2, 2020, Pages 491-510, ISSN 0377-2217, https://doi.org/10.1016/j.ejor.2019.11.013.

[5] Mpofu, K.; Tlale, N.S. Multi-level decision making in reconfigurable machining systems using fuzzy logic, Journal of Manufacturing Systems, Volume 31, Issue 2, 2012, Pages 103-112, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2011.08.006.

[6] Chen Zheng, Xiansheng Qin, Benoît Eynard, Jing Bai, Jing Li, Yicha Zhang, SME-oriented flexible design approach for robotic manufacturing systems, Journal of Manufacturing Systems, Volume 53, 2019, Pages 62-74, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2019.09.010.

[7] Saliba, M. A.; Zammit, D.; Azzopardi, S. Towards practical, high-level guidelines to promote company strategy for the use of reconfigurable manufacturing automation, Robotics and Computer-Integrated Manufacturing, Volume 47, 2017, Pages 53-60, ISSN 0736-5845, https://doi.org/10.1016/j.rcim.2016.12.002.

[8] Ferreras-Higuero, E.; Leal-Muñoz, E.; García de Jalón, J.; Chacón, E.; Vizán, A. Robot-process precision modelling for the improvement of productivity in flexible manufacturing cells, Robotics and Computer-Integrated Manufacturing, Volume 65, 2020, 101966, ISSN 0736-5845, https://doi.org/10.1016/j.rcim.2020.101966.

[9] Kontovourkis, O.; Phocas, M. C.; Katsambas, C. Digital to physical development of a reconfigurable modular formwork for concrete casting and assembling of a shell structure, Automation in Construction, Volume 106, 2019, 102855, ISSN 0926-5805, https://doi.org/10.1016/j.autcon.2019.102855.

[10] Björnsson,A.; Jonsson,M.; Johansen,K. Automated material handling in composite manufacturing using pick-and-place systems – a review, Robotics and Computer-Integrated Manufacturing, Volume 51, 2018, Pages 222-229, ISSN 0736-5845, https://doi.org/10.1016/j.rcim.2017.12.003.

[11] M.G. Abou-Ali, M.A. Shouman, Effect of dynamic and static dispatching strategies on dynamically planned and unplanned FMS, Journal of Materials Processing Technology, Volume 148, Issue 1, 2004, Pages 132-138, ISSN 0924-0136, https://doi.org/10.1016/j.jmatprotec.2004.01.054.

[12] Jason Y.K. Chan, et Al. Prospective clinical trial to evaluate safety and feasibility of using a single port flexible robotic system for transoral head and neck surgery, Oral Oncology, Volume 94, 2019, Pages 101-105, ISSN 1368-8375, https://doi.org/10.1016/j.oraloncology.2019.05.018.

[13] Michael Z. Lerner, Michael Tricoli, Marshall Strome, Abrasion and blunt tissue trauma study of a novel flexible robotic system in the porcine model, American Journal of Otolaryngology, Volume 38, Issue 4, 2017, Pages 447-451, ISSN 0196-0709, https://doi.org/10.1016/j.amjoto.2017.04.002.

[14] Jair Carlos Dutra, et Al. Development of a flexible robotic welding system for weld overlay cladding of thermoelectrical plants' boiler tube walls, Mechatronics, Volume 24, Issue 5, 2014, Pages 416-425, ISSN 0957-4158, https://doi.org/10.1016/j.mechatronics.2014.03.002.

[15] Forbes, J.; Damaren, C. Design of optimal strictly positive real controllers using numerical optimization for the control of flexible robotic systems, Journal of the Franklin Institute, Volume 348, Issue 8, 2011, Pages 2191-2215, ISSN 0016-0032, https://doi.org/10.1016/j.jfranklin.2011.06.013.

[16] Gabriel G. Kost, Ryszard Zdanowicz, Modeling of manufacturing systems and robot motions, Journal of Materials Processing Technology, Volumes 164–165, 2005, Pages 1369-1378, ISSN 0924-0136, https://doi.org/10.1016/j.jmatprotec.2005.02.186.

[17] Giulio Rosati, Simone Minto, Fabio Oscari, Design and construction of a variable-aperture gripper for flexible automated assembly, Robotics and Computer-Integrated Manufacturing, Volume 48, 2017, Pages 157-166, ISSN 0736-5845, https://doi.org/10.1016/j.rcim.2017.03.010.

[18] Yin, G. et Al. Flexible punching system using industrial robots for automotive panels, Robotics and Computer-Integrated Manufacturing, Volume 52, 2018, Pages 92-99, ISSN 0736-5845, https://doi.org/10.1016/j.rcim.2017.11.002.

[19] G. Nagamani, Young Hoon Joo, G. Soundararajan, Reza Mohajerpoor, Robust event-triggered reliable control for T-S fuzzy uncertain systems via weighted based inequality, Information Sciences, Volume 512,

[20] 2020, Pages 31-49, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2019.09.034.

[21] Qiao G, Lu RF, McLean C. Flexible manufacturing systems for mass customisation manufacturing. International Journal of Mass Customisation. 2006 Jan 1;1(2-3):374-93.

[22] Zheng, C. et Al. Survey on Design Approaches for Robotic Manufacturing Systems in SMEs, Procedia CIRP, Volume 84, 2019, Pages 16-21, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2019.04.183.

[23] Wilms, M. et Al. Development of a decision logic for the selection of a flexible robotic system for the automated manufacturing in tooling, Procedia CIRP, Volume 81, 2019, Pages 435-440, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2019.03.075.

[24] Ajeil, F. ; et al. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm, Applied Soft Computing **2020**, 89, Article 106076

[25] Seeja, G.; et al. A Survey on Swarm Robotic Modeling, Analysis and Hardware Architecture. Procedia Computer Science **2018** , 133, 478-485

[26] Erdem, O.; Carus, A.; Erdem, H.; Carus, A.; Le, H .Large-scale SRAM-based IP lookup architectures using compact trie search structures. Computers & Electrical Engineering **2014**, 40, 1186-1198.

[27] Graaf, B.; Weber, S.; Deursen, A. Model-driven migration of supervisory machine control architectures. Journal of Systems and Software **2008**, 81, 517-535.

[28] Yulan, H. ; Qisong, Z.; Pengfei, X. Study on Multi-Robot Cooperation Stalking Using Finite State Machine. Procedia Engineering **2012**, 29, 3502-3506.

[29] Dai, Y.; et al. A switching formation strategy for obstacle avoidance of a multi-robot system based on robot priority model. ISA Transactions **2015** , 56, 123-134

[30] Kuhner, D.; et al. A service assistant combining autonomous robotics, flexible goal formulation, and deep-learning-based brain–computer interfacing. Robotics and Autonomous Systems **2019**, 116, 98-113

[31] Romero, A.; et al. Simplifying the creation and management of utility models in continuous domains for cognitive robotics. Neurocomputing **2019**, 35311, 106-118.

[32] Lemaignan, S.; Warnier, M. ; Sisbot, E. ; Clodic, A. ; Alami, R. Artificial cognition for social human–robot interaction: An implementation Artificial Intelligence **2017**, 247, 45-69.

[33] Baklouti, E.; Ben Amor, N.; Jallouli, M. Reactive control architecture for mobile robot autonomous navigation, Robotics and Autonomous Systems, Volume 89, 2017, Pages 9-14, ISSN 0921-8890, https://doi.org/10.1016/j.robot.2016.09.001.

[34] Yu, C.; et al. Onboard system of hybrid underwater robotic vehicles: Integrated software architecture and control algorithm. Ocean Engineering **2019**, 187, Article 106121.

[35] Gharbi, A. A Social Multi-Agent Cooperation System Based on Planning and Distributed Task Allocation, Information, 11(5) 271; doi:10.3390/info11050271 (2020).

[36] Gharbi, A. Five Capabilities Model Applied to Multi-Robot Systems. International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC), 7(1), pp.57-88, 2015

[37] Gharbi, A.; Gharsellaoui, H.; Ben Ahmed, S. Multi-Agent Control System, ICSOFT, EA, 2014:117-124.

[38] Au, T.C., Kuter, U. and Nau, D., 2008, May. Planning for interactions among autonomous agents. In International Workshop on Programming Multi-Agent Systems (pp. 1-23). Springer, Berlin, Heidelberg.

[39] Anunciene Barbosa Duarte, et al. Genetic diversity between and within full-sib families of Jatropha using ISSR markers, Industrial Crops and Products, Volume 124, 2018, Pages 899-905, ISSN 0926-6690, https://doi.org/10.1016/j.indcrop.2018.08.066.