Robotics and Computer-Integrated Manufacturing 33 (2015) 42–55

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Towards robust assembly with knowledge representation for the planning domain definition language (PDDL)

Z. Kootbally [a,*], C. Schlenoff [b], C. Lawler [a], T. Kramer [c], S.K. Gupta [d]

[a] Department of Mechanical Engineering, University of Maryland, College Park, MD 20740, USA
[b] Intelligent Systems Division, National Institute of Standards and Technology, Gaithersburg, MD, USA
[c] Department of Mechanical Engineering, Catholic University of America, Washington, DC, USA
[d] Maryland Robotics Center, University of Maryland, College Park, MD, USA

ARTICLE INFO

Article history:
Received 1 May 2014
Received in revised form
31 July 2014
Accepted 11 August 2014
Available online 10 September 2014

Keywords:
PDDL (Planning Domain Definition Language)
Planning
Replanning
Agility
Knowledge representation
Robotics

ABSTRACT

The effort described in this paper attempts to integrate agility aspects in the "Agility Performance of Robotic Systems" (APRS) project, developed at the National Institute of Standards and Technology (NIST). The new technical idea for the APRS project is to develop the measurement science in the form of an integrated agility framework enabling manufacturers to assess and assure the agility performance of their robot systems. This framework includes robot agility performance metrics, information models, test methods, and protocols. This paper presents models for the Planning Domain Definition Language (PDDL), used within the APRS project. PDDL is an attempt to standardize Artificial Intelligence planning languages. The described models have been fully defined in the XML Schema Definition Language (XSDL) and in the Web Ontology Language (OWL) for kit building applications. Kit building or kitting is a process that brings parts that will be used in assembly operations together in a kit and then moves the kit to the area where the parts are used in the final assembly. Furthermore, the paper discusses a tool that is capable of automatically and dynamically generating PDDL files from the models in order to generate a plan or to replan from scratch. Finally, the ability of the tool to update a PDDL problem file from a relational database for replanning to recover from failures is presented.

1. Introduction

The new technical idea for the "Agility Performance of Robotic Systems" (APRS) project [1] at the National Institute of Standards and Technology (NIST) is to develop the measurement science in the form of an integrated agility framework enabling manufacturers to assess and assure the agility performance of their robot systems. This framework includes robot agility performance metrics, information models, test methods, and protocols – all of which are validated using a combined virtual and real testing environment. The information models enumerate and make explicit the necessary knowledge for achieving rapid re-tasking and being agile and will answer question such as "What does the robot need to know?", "When does it need to know it?", and "How will it get that knowledge?". This framework will (1) allow manufacturers to easily and rapidly reconfigure and re-task robot systems in assembly operations, (2) make robots more accessible to small and medium organizations, (3) provide large organizations greater efficiency in their assembly operations, and (4) allow the US to compete effectively in the global market. Any company that is currently deploying or planning to deploy robot systems will benefit because it will be able to accurately predict the agility performance of its robot systems and be able to quickly re-task and reconfigure its assembly operations.

The increased number of new models and variants has forced manufacturing firms to meet the demands of a diversified customer base by creating products in a short development cycle, yielding low cost, high quality, and sufficient quantity. Modern manufacturing enterprises have two alternatives to face the aforementioned requirements. The first one is to use manufacturing plants with excess capacity and stock of products in inventory to smooth fluctuations in demand. The second one is to use and increase the flexibility of their manufacturing plants to deal with the production volume and variety. While the use of flexibility generates the complexity of its implementation, it still is the preferred solution. Chryssolouris [2] identified manufacturing flexibility as an important attribute to overcome the increased

* Corresponding author.
E-mail addresses: zeid.kootbally@nist.gov (Z. Kootbally), craig.schlenoff@nist.gov (C. Schlenoff), crlawler@umd.edu (C. Lawler), thomas.kramer@nist.gov (T. Kramer), skgupta@umd.edu (S.K. Gupta).

http://dx.doi.org/10.1016/j.rcim.2014.08.006
0736-5845/

number of new models and variants from customized demands. Flexibility, however needs to be defined in a quantified fashion before being considered in the decision making process.

Agility is often perceived as combination of speed and flexibility. Gunasekaran [3] defines agile manufacturing as the capability to survive and prosper in a competitive environment of continuous and unpredictable change by reacting quickly and effectively to changing markets, driven by customer-designed products and services. To be able to respond effectively to changing customer needs in a volatile marketplace means being able to handle variety and introduce new products quickly. Lindbergh [4] and Sharafi and Zhang [5] mentioned that agility consists of flexibility and speed. Essentially, an organization must be able to *respond flexibly* and *respond speedily* [6]. Conboy and Fitzgerald [7] identified terms such as *speed* [8], *quick* [9–12], *rapid* [13], and *fast* [14] that occur in most definitions of agility.

The above definitions of agile manufacturing can be applied at the assembly level of a manufacturing system. The assembly system needs to have a certain level of flexibility in the presence of disturbances that can be expressed by the degree of robustness. Kannan and Parker [15] described robustness as the ability of the system to identify and recover from faults. Robustness of a control system was described by Leitão [16] as the capability to remain working correctly and relatively stable, even in the presence of disturbances. The concept of robustness discussed in this paper is expressed with replanning and plan repair for failure recovery (e.g., misalignments, incorrect parts and tooling, shortage of parts, or missing tool). Fox et al. [17] discussed replanning and plan repair when differences are detected between the expected and the actual context of execution during plan execution in real environments. The latter authors define plan repair as the work of adapting an existing plan to a new context while perturbing the original plan as little as possible. Replanning is defined as the work of generating a new plan from scratch.

This paper first describes the models developed to represent structures of the planning language in the APRS project. The APRS project is working in collaboration with the IEEE Robotics and Automation Society's Ontologies for Robotics and Automation (ORA) Working Group to develop information models related to kitting [18–21], including a model of the kitting environment and a model of a kitting plan. Kitting or kit building is the process in which several different, but related items are placed into a container and supplied together as a single unit. Kitting itself may be viewed as a specialization of the general bin-picking problem [22,23]. In industrial assembly of manufactured products, kitting is often performed prior to final assembly. Manufacturers utilize kitting due to its ability to provide cost savings [24] including saving manufacturing or assembly space [25], reducing assembly workers walking and searching times [26], and increasing line flexibility [27] and balance [28]. It is anticipated that utilization of the knowledge representation will allow for the development of higher performing kitting systems and will lead to the development of agile automated robot assembly.

Planning for kitting relies on the Planning Domain Definition Language (PDDL) [29]. In order to operate, the PDDL planners require a PDDL file-set that consists of two files that specify the domain and the problem. From these files, the planning system creates an additional static plan file. Structures of PDDL domain and problem files are fully defined in each of two languages: XML Schema Definition Language (XSDL) [30] and Web Ontology Language (OWL) [31]. Furthermore, this paper describes a tool that is capable of automatically and dynamically generating PDDL domain and problem files from the OWL models. The tool is also used to repair a PDDL problem file in order to replan from failures.

This paper is structured as follows: an overview of the knowledge driven methodology for the APRS project is presented in Section 2. The XSDL models that were developed to represent PDDL domain and problem files are discussed in Section 3. A tool that is capable of (1) dynamically producing PDDL domain and problem files from OWL files and (2) updating PDDL problem files from a dynamic relational database is described in Sections 4 and 5, respectively. Finally, concluding remarks and future work are addressed in Section 6.

2. Knowledge driven methodology

The knowledge driven methodology presented in this section is not intended to act as a stand-alone system architecture. Rather it is intended to be an extension to well-developed hierarchical, deliberative architectures such as 4D/RCS [32]. The overall knowledge driven methodology of the system is depicted in Fig. 1. Although the described architecture is currently used in a simulation environment, its application can be extended to a real environment. The remainder of this section gives a brief description of the components pertaining to the effort presented in this paper.

- Use Case Scenarios: At the early stage of assembly, new orders coming from customers are entered in the system by an operator via a graphical user interface. The information that is required in this step is for instance the type of assembly and the number of products required. This first step is therefore an attempt to introduce agility in the system with a functionality that smooths fluctuations in demand.
- Knowledge (OWL/XML): At the next level up, the information encoded in the Use Case Scenarios is then organized into a domain independent representation. The Knowledge (OWL/XML) component contains all the basic information that was determined to be needed during the evaluation of the Use Case Scenarios. This component consists of class files and instance files that describe the environment (Environment), including the initial (Initial Conditions) and goal (Goal Conditions) states for the current assembly, and PDDL actions (SOAP). The knowledge is represented in a compact form with knowledge classes inheriting common attributes from parent classes. The SOAP knowledge describes aspects of PDDL actions that are required for the domain under study. The instance files describe the initial and goal states for the system through the Initial Conditions file and the Goal Conditions file, respectively. The initial state file must contain a description of the environment that is complete enough for a planning system to be able to create a valid sequence of actions that will achieve the given goal state. The goal state file only needs to contain information that is relevant to the end goal of the system.
  Since both the OWL and XML implementations of the knowledge representation are file based, real time information proved to be problematic. In order to solve this problem, an automatically generated MySQL database has been introduced as part of the knowledge representation. Different frameworks (e.g., Jena [33]) are capable to store ontologies in memory, however, the particularity of a MySQL database is that it allows information of the environment to be shared between multiple robots in the case of collaborative kitting.
- Planning: At the next level up, aspects of this knowledge are automatically extracted and encoded in a form that is optimized for a planning system to utilize. The planning language used in the knowledge driven system is PDDL. The PDDL input format consists of two files that specify the domain and the problem. As shown in Fig. 1, these files are automatically generated from a set of OWL files. The PDDL Domain file is