

Journal of Robotics and Mechanical Engineering Research

A Disassembly Planning Information Model for Adaptive Disassembly Planning

Bicheng Zhu* and Utpal Roy

Syracuse University

*Corresponding author: Bicheng Zhu, Syracuse University; E mail: bizhu@syr.edu

Article Type: Research, **Submission Date:** 23 July 2018, **Accepted Date:** 14 August 2018, **Published Date:** 13 September 2018.

Citation: Bicheng Zhu and Utpal Roy (2018) A Disassembly Planning Information Model for Adaptive Disassembly Planning. J Robot Mech Eng Resr 2(3): 8-23. doi: <https://doi.org/10.24218/jrmer.2018.28>.

Copyright: © 2018 Bicheng Zhu and Utpal Roy. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract

This work solves the adaptive disassembly planning problem based on a disassembly sequence generator Information Model, which is a sub model belonging to a disassembly information base called Disassembly Information Model (DIM). DIM is developed based on an extensive investigation of various informational aspects in the domain of disassembly planning and represents an appropriate systematization and classification of the products, processes, uncertainties and degradations related information. In this paper, the goal is to partially validate the DIM model by using it for solving the adaptive disassembly planning problem.

Introduction

Disassembly, as the core step in the EOL product recovery, is defined as “A systematic method for separating a product into its constituent parts, components and subassembly” [1]. Both the potential economic profits and the regulatory laws motivate the study of the EOL product disassembly modeling and implementation and carrying out the disassembly process “optimally” plays a critical role in the entire process of the EOL product recovery. Over the years, various methods ranging from network theory to mathematical programming have been applied in the domain of product disassembly [2]. Unfortunately, very few research looks into the problem from the information aspect, which in the author’s opinion, is the bottleneck of the current disassembly related research. In detail, the challenge is that disassembly planners have limited knowledge on what information is critical in the planning of the disassembly process, how to access this information, and, finally how to utilize the updated on-site information (which is unknown in the beginning of the disassembly process) for dynamically adapting the “optimal” disassembly process plan. Also, an EOL product is highly independent and has to be treated individually, which further aggravates the above mentioned problems.

Fortunately, with the advent of the internet, disassembly research

has the opportunity to leap forward and overcome the obstacles mentioned above. Two specific thriving technologies under the umbrella of smart manufacturing, such as Internet of Things (IoT) and Life Cycle Units (LCU), have already been discussed in the disassembly research community for ideas like future cloud-based remanufacturing [3] and semantic recovery information service [4]. Briefly, IoT provides a network to connect different physical objects, which allows them to be sensed and controlled remotely across existing network infrastructure, creating opportunities for more direct integrations of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit. LCU, on the other hand, is developed specifically for the product disassembly process. As mentioned before, in a disassembly factory, different products arrive continuously for disassembly, and individual decisions regarding optimal disassembly sequences have to be made for every product. It is difficult to predict any pre-defined disassembly process sequences a priori, so the detailed information on how to disassemble each arriving product is needed. LCU is proposed under the idea of decentralizing that information by integrating a physical device named Life Cycle Units (LCU) into every product. The LCU stores information needed for disassembly. Once enough disassembly information about a product is present, the optimal disassembly sequence can be generated based on the actual physical status of the EOL product. Combining the LCU and IoT technologies together, individualized EOL product information could be sensed and collected by LCU and transferred to the central Product Lifecycle Management (PLM) system through the IoT network. Now, disassembly researchers could have the potentials to tackle the problem of disassembly information bottleneck.

In this paper, we developed a Disassembly Information Model (DIM) that can be integrated into the current Smart Manufacturing paradigm for efficient disassembly planning activities. Disassembly planning related information are

identified and generalized through extensive literature reviews, and they can be partitioned into relevant sub models. A layered Information Model (IM) development methodology is proposed to address the reusability-usability trade-off problem. The developed DIM is further implemented into the Web Ontology Language (OWL), though which relevant information can be computational analyzed and utilized. The validation of the DIM model is carried out through disassembly planning related application developments and one prototype application called “Adaptive Disassembly Planner” is developed in this work.

Literature Review

The Information Model, sometimes called ontology, is the consensual modelling of concepts and relationship in a domain of interest. Over the years, researchers have contributed to the development of IM or ontology in the domain of manufacturing, with different focusing aspects. Some notable work is reviewed below.

Leimagnan et al. [5] developed the Manufacturing Semantic Ontology (MASON) to formally capture the concepts related to the manufacturing industry. The semantics related to entity, resources and operation were captured in formal logic using web ontology language (OWL). Two applications about automatic cost estimation and the semantic-aware multi-agent system for manufacturing were discussed to demonstrate the usefulness of the proposed MASON ontology.

Xiaomeng [6] selected the field of Design for Manufacturing (DFM) for his PhD study and three primary aspects are investigated. First, a generalized DFM ontology is proposed and developed, which fulfills the mathematical and logical constraints needed in the domain of DFM. Second, the means to guide users to the proper information and integrate heterogeneous data resources is investigated. Third, a decision support tool is developed to help designers consider the design problem in a systematic way based on the developed DFM ontology.

Pavan [7] developed an ontology called the Design Activity Ontology (DAO) to explicitly represent the design activity that can cover phases of the design process from conceptual phase through detail design phase. The ontology provides a formalized and structured vocabulary of design activities for the exchange of design process models and it further enables design processes to be modeled, analyzed and optimized in a consistent way.

Kim et al. [8] proposed a collaborative assembly design framework that offers a shared conceptualization of assembly modeling and an Assembly Design Ontology (AsD) is developed to capture the joining intents of a product. AsD is claimed to serve as a formal, explicit specification of assembly design so that it makes the assembly knowledge both machine-interpretable and sharable.

Some industrial efforts have also been devoted to the development of the manufacturing related Information Model, a notable development in this field is led by NIST. One of their work is the NIST’s Core Product Model (CPM), which a unified

modeling language (UML) based model intended to capture the full range of engineering information commonly shared in product development [9]. CPM focuses on modeling the general, common and generic product information and excludes the information which is domain specific. NIST further developed another information model called “Open Assembly Model” (OAM) [10] which extends CPM. Along with the structural information, it represents the function, form, and behavior of the assembly, and defines a system level conceptual model.

Recently, NIST also proposed a disassembly information model [11] and to the author’s knowledge, this is the first attempt to develop disassembly related information model. However, the focus of the NIST disassembly information model is on the reuse, maintenance, and recycling aspect and the information regarding to the disassembly planning is not well addressed. Also, the reusability/usability tradeoff problem is not considered in the NIST disassembly information model. Lastly, the NIST disassembly information model remains in the conceptual stage and the implementation under the paradigm of smart product and IoT has not been carried out.

Overview of Disassembly Planning Information Model (DIM)

Disassembly information requirement analysis has been carried out by the author’s previous research [12]. In general, DIM should comprise of the information related to the aspects of product, process, uncertainty and degradation and the modelling of which involves certain information modeling patterns like n-ary relationship, part-whole relationship, etc. On the other hand, DIM should also achieve certain balances between IM usability and reusability. Thus, a layered modelling methodology has been proposed, in which DIM has been subdivided by means of layers (Figure 1), with intention to separate general knowledge into different level of abstractions. Also, a “minimal ontological commitment” [13] guideline is followed, which means each layer holds only concepts/relationships and axioms that are essential for the function of the current layer. Information that is not essential for the layer’s purpose are sourced out to lower layers. Details of each layer are presented as follows:

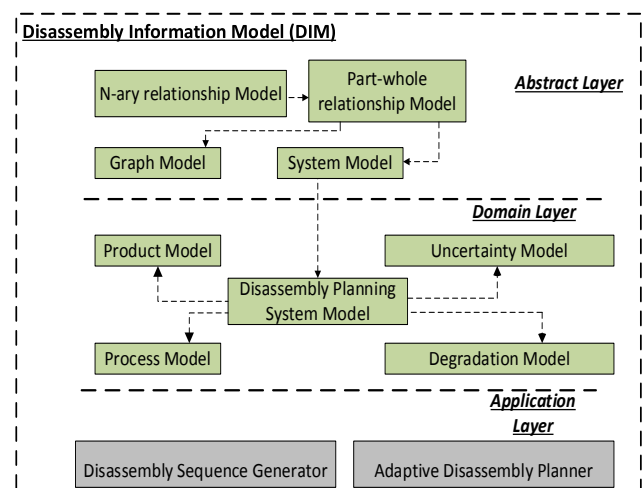


Figure 1: The Overall Structure of DIM

Abstract Layer: The Information Models in the abstract layer hold the fundamental modeling concepts, which are independent of a particular problem or domain and can therefore be universally applied. They describe the design guidelines (design pattern) for the construction of the other sub models in the DIM. Models like n-ary relationship, part-whole relationship, graph model and system model belong to this layer.

Domain Layer: The Information Models in the domain layer capture the knowledge related to a domain of expertise, such as disassembly planning in our case, and they generally don't target on solving a specific problem or task, but rather providing a domain knowledge foundation for a range of different applications. Thus, the Information Model residing on this layer is more specific than those in the abstract layer, but less specific than those in the lower layer (application layer). The majority of the required disassembly domain information (product, process, etc.) are implemented in the models in this layer.

Application Layer: represents the most specific Information Model which is directly usable for a certain disassembly planning application. This paper focuses on two disassembly planning applications: (1) Disassembly Sequence Generator and (2) Adaptive Disassembly Planning.

Such a layered DIM development methodology takes the IM reusability-usability trade-off problem into account. The abstract or general knowledge is modeled in the sub models located on the top layer of the DIM. They provide various design patterns which can be reused in various application contexts and normally are not directly usable due to the high abstraction. On the other hand, knowledge in the models residing on the lower layer is ready to be used, but is usually application specific and thus hardly to be transferred to other applications. Information Models in each layer of the DIM contain knowledge with certain degrees of reusability and usability and the usability of the knowledge normally increases with descending reusability when navigating from the top to the bottom layers of DIM.

In the following sections, DIM sub model "Adaptive Disassembly Sequence Planning Information Model" in the application layer is presented in detail, other sub models in the abstract and domain can be found at Zhu [12].

Adaptive Disassembly Planning Information Model

This section presents the details related to the Adaptive Disassembly Planning Information Model. We start with the adaptive disassembly planning problem description in section 5.1. Next, the requirement for the Adaptive Disassembly Planning Information Model, residing on the application layer of the developed DIM, is presented in detail in section 5.2. Section 5.3 presents detailed Adaptive Disassembly Planning Information Model using the UML class diagram as a graphical notation. The detailed application algorithm for carrying out the sequence generation and optimization process is presented in section 5.4. Lastly, section 5.5 verify the application procedure with a kitchen exhaust product as a case study.

Adaptive Disassembly Planning Problem Description

Adaptive disassembly planning takes all the feasible disassembly sequences as input and targets on locating the optimal disassembly sequence with consideration of two extra issues as follows:

(1) **Uncertainty issue:** different from the assembly process, the disassembly process has various uncertainty issues in nature. Thus, extra information and special mechanisms are needed for such uncertainty handling. Two types of uncertainties need to be addressed in the disassembly planning: (1) Component/assembly function uncertainty and (2) Operation uncertainty.

Component/assembly function uncertainty: each component or assembly might associate with a primary function, which contributes to the product overall function. Such function may not be working when the EOL product becomes obsolete. Such functioning/non-functioning information is critical in the disassembly planning process and can only be revealed gradually during the disassembly process.

Operation uncertainty: during the disassembly process, certain operation, such as unscrewing, might not success due to the bad component condition like deformation or corrosion. Then, extra special operations are necessary to handle such situations, which will incur a higher cost. Since this information is also unknown at the beginning of the disassembly process, it is called operation uncertainty.

(2) **Degradation issue:** Component/assembly degradation is also a critical issue in the planning of disassembly. Degradation is a gradual change in the properties (like tensile strength, color, shape, etc.) of the component, which usually does not affect the overall function of a component until reaching a critical point. However, degradation does affect the economic quantification of EOL product or component. For example, some subassembly might work fine (functional) after the function testing, but the reuse value of the subassembly still could be lower than the expected average reuse value (the subassembly is close to failure) or higher than the expected average reuse value (the subassembly still has a long remaining useful life time).

In order to handle the above two issues, extra information is needed and has been identified in DIM domain layer sub models (the Uncertainty Information Model and the Degradation Information Model) (Zhu July, 2016) . However, the determination of the value of some of this information for a specific EOL product can hardly be done a priori (e.g. the condition of an internal component usually cannot be identified at the beginning of the disassembly process). Rather, they are revealed gradually with the disassembly process. Thus, an "optimal" path is determined at each stage of the disassembly process with the limited information supplied and will be re-evaluated after reaching a new stage with more information identified. Thus, it is called adaptive disassembly planning problem.

Adaptive Disassembly Planning Information Requirement Analysis

The required information for handling the uncertainty and degradation issue has been identified in the domain level sub models, named the Uncertainty Model and the Degradation Model. The Uncertainty Model is based on the Bayesian Network theory, whereas the Degradation Model is based on the Fuzzy Logic theory [12]. From a high level view, the main information in both of the models is basically probability theory based statistical information, which provides certain degrees of belief in the relevant issues. However, in order to make disassembly decision, disassembly benefits (utility) and disassembly constraints should also be considered and a certain disassembly decision theory should be modeled. In this work, disassembly decision theory is defined as:

Disassembly Decision Theory = Probability Theory + Utility Theory + Disassembly Constraints

The fundamental idea of the disassembly decision theory is that a computer aided disassembly planner is rational if and only if it chooses the feasible disassembly action (satisfying all the constraints) that yields the highest expected disassembly utility, averaged over all the possible outcomes of the action. This is also called the principle of Maximum Expected Utility (MEU) in the traditional decision theory.

The realization of the Disassembly Decision Theory yields what we called Disassembly Decision Network (DDN) and it can be described formally as a six-tuple: $DDN = (P-DN, UTN, UN, TR, CPT, F)$, where

Process Decision Node (P-DN): $P-DN = \{P-DN_1, P-DN_2, P-DN_3, \dots, P-DN_N\}$, $N > 0$, is a finite set of process decision nodes denoted by rectangle shape. Each of the node can take two possible values ("carry out" or "do not carry out"), which represents the two choices available to the disassembly process planner regarding to a specific process decision.

Utility Node (UTN): $UTN = \{UTN_1, UTN_2, UTN_3, \dots, UTN_N\}$, $N > 0$, is a finite set of utility nodes denoted by diamond shape and they are used to enable the numerical evaluation of decision consequences. Two types of the utility nodes are further specified:

Process Utility Node (P-UTN): represents how much cost is associated with a disassembly process.

Disassembly Object Utility Node (D-UTN): represents how much utility is associated with a disassembly object, like component, subassembly, etc. The utility can be interpreted as reuse value, recycling value or discard cost depending on the disassembly context (type of the disassembly object, whether or not the component is functioning, whether or not the subassembly is further detached, etc.)

Uncertainty Node (UN): $UN = \{UN_1, UN_2, UN_3, \dots, UN_N\}$, $N > 0$, is a finite set of uncertainty or chance nodes denoted by ellipse shape and they are used to represent the random variables related to

the problem. Two types of the uncertainty nodes are further specified:

Process Uncertainty Node (P-UN): represents whether a disassembly process is successfully carried out and two values are possible for this type of uncertainty node: {"success", "fail"}.

Disassembly Object Function Uncertainty Node (D-UN): represents whether a disassembly object is performing its designed function properly and two values are possible for this type of uncertainty node: {"function", "not function"}.

Transition Arc (TR): $TR = \{TR_1, TR_2, TR_3, \dots, TR_N\}$, $N > 0$, is a finite set of directed arcs connecting different types of nodes. The intuitive meaning of a transition arc from node X to node Y is that X has a direct influence on Y, or there exists a causal relationship between X (cause) and Y (effect). Based on the types of the nodes to be connected, five types of TRs are further specified as follows:

Type 1 ($P-DN \rightarrow P-UTN$): A transition arc connecting from a P-DN to a P-UTN, which describes the influences of a process decision on the process utility. In general, if the decision of a certain disassembly process is "carry out", then the utility (cost) of the relevant process is set to some negative value. On the other hand, if the decision of a certain disassembly process is "do not carry out", the relevant process utility (cost) should be zero.

Type 2 ($P-UN \rightarrow P-UTN$): A transition arc connecting from a P-UN to a P-UTN, which describes the effect of the process uncertainty on the process utility. In general, if the disassembly process is successfully executed without problem (the value of P-UN is "success"), the process utility (cost) will be set to the average process cost. On the other hand, if the disassembly process fails, a higher process utility (cost) will be applied.

Type 3 ($D-UN \rightarrow D-UTN$): A transition arc connecting from a D-UN to a D-UTN, which describes the effect of the disassembly object function uncertainty on the disassembly object utility. In general, if the disassembly object is "not functioning", it means this disassembly object cannot be reused and thus the relevant utility is set to either the recycle value (if it is a component) or discard cost (if it is a subassembly). On the other hand, if it is "functioning", it means the disassembly object can be reused and the relevant utility should be set to the reuse value.

Type 4 ($P-DN \rightarrow D-UTN$): A transition arc connecting from a P-DN to a D-UTN, which describes the influences of a process decision on the disassembly object utility. In general, if a process disassembles the disassembly object, then the relevant utility is set to zero (the disassembly object doesn't exist anymore). Otherwise, the relevant utility will be set to either the reuse value, or the recycling value/the discard cost depending whether the disassembly object is functioning properly.

Type 5 ($D-UN \rightarrow D-UN$): A transition arc connecting from a D-UN to a D-UN, which describes the function dependency between different disassembly objects. As an example, whether or not a computer is functioning properly is dependent on the functionality of its internal component, like CPU, motherboard,

etc. It can be represented as: $D-UN_{\text{motherboard}} \rightarrow D-UN_{\text{CPU}} \rightarrow D-UN_{\text{Computer}} \rightarrow D-UN_{\text{Computer}}$.

Conditional Probability Table (CPT): $CPT = \{CPT_1, CPT_2, CPT_3, \dots, CPT_N\}$, $N > 0$, is a finite set of conditional probability tables and each which is attached an uncertainty node described above. For each node, a CPT represents the conditional probability distribution, which quantifies the effect of the parents on the node. This is the statistical information, which has been included in the Uncertainty Model in the domain layer of DIM.

Fuzzy model (F): $F = \{F_1, F_2, F_3, \dots, F_N\}$, $N > 0$, is a finite set of fuzzy models and each which is attached to a Disassembly Object Utility Node (D-UTN) described above. It is used to quantify the degradation of disassembly objects by evaluating its real reuse value. The detail information related to the fuzzy model has been included in the Degradation Model in the domain layer of DIM.

To summarize, the requirement for the Adaptive Disassembly Planning Information Model is to provide information elements to support the construction of the Disassembly Decision Network described above. Also, some information has been modeled in the domain layer sub models like Process Model, Uncertainty Model and Degradation Model. Thus, an integration of these models is also needed. Table 1 summarizes the modeling requirements for the Adaptive Disassembly Planning Information Model.

Table 1: Requirements for the Adaptive Disassembly Planning Information Model

R1	The modeling of different types of node: decision node, uncertainty node and utility node
R2	The modeling of five different types of transition arc.
R3	The linking to the uncertainty model and degradation model for the retrieval of relevant statistical information.

Formal Adaptive Disassembly Planning Information Model

The Adaptive Disassembly Planning Information Model deals with the information required for the adaptive disassembly planning problem. It is residing on the application layer of the DIM and is being extended based on three domain layer sub models named Process model, Uncertainty Model and Degradation Model. The overall structure is shown in Figure 2 below. We will describe the model according to the information requirements identified above in the following sections.

R1->Node Modelling: Based on the definition of the Disassembly Decision Network, five classes representing different types of nodes have been modeled in the Adaptive Disassembly Planning Information Model: (1) class **Process_Decision_Node** representing P-DN, (2) class **Process_Utility_Node** representing P-UTN, (3) class **Process_Uncertainty_Node** representing P-UN, (4) class **Disassembly_Object_Function_Uncertainty_Node** representing D-UN, and (5) class **Disassembly_Object_Utility_Node** representing D-UTN.

R2->Transition Arc Modelling: Influences exists between different types of nodes and each of which form a certain transition arc in the Disassembly Decision Network. From the requirement analysis carried out before [12], five types of transition arcs are identified and they are being implemented in the Adaptive Disassembly Planning Information Model by introducing the object property “influence”, which connects two nodes as its domain object and range object.

As shown in Figure 2, three types of transition arcs (Type 1 to Type 3) have been explicitly defined. As an example, type 1 transition arc indicates thecausal effect of a process decision

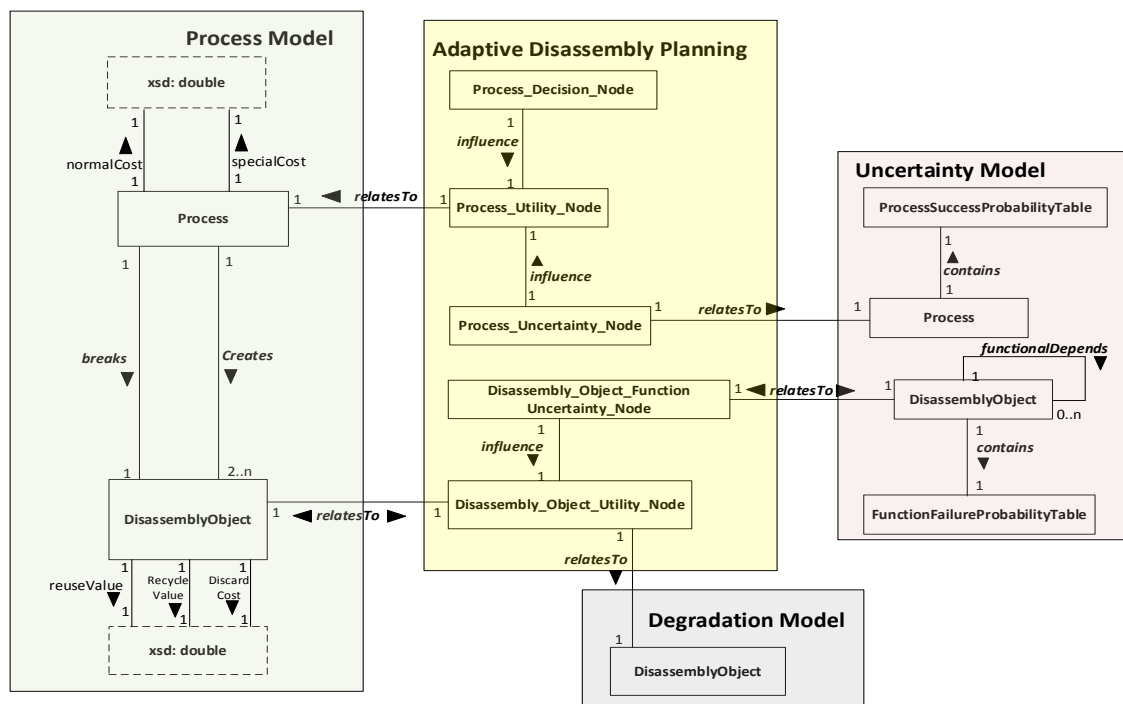


Figure 2: The Adaptive Disassembly Planning Information Model

(**Process_Decision_Node**) on the process utility (**Process_Utility_Node**). The two relevant nodes are related through the object property “*influence*”: the domain of the “*influence*” object property is the **Process_Decision_Node** class, which indicates the causes, whereas the range of the “*influence*” object property is the **Process_Utility_Node** class, which indicates the effects of the causes.

A similar approach can be used to model Type 4 and Type 5 transition arc by introducing the “*influence*” object property to link from the **Process_Decision_Node** class to the **Disassembly_Object_Utility_Node** class (Type 4 ($P-DN \rightarrow D-UTN$)); or to link from the **Disassembly_Object_Utility_Node** class to the **Disassembly_Object_Utility_Node** class (Type 5 ($D-UN \rightarrow D-UN$)). However, such approach will yield redundant or duplicate information due to the fact that the Type 4 and Type 5 transition information have already been implicitly indicated in the domain level Process Model and the Uncertainty Model. Thus, we utilize semantic rules to transfer such implicit information in the Process Model and Uncertainty Model to the explicit Type 4 and Type 5 causal information, which can be utilized to construct the Disassembly Decision Network. The detail modeling mechanism is presented below:

Modeling of Type 4 ($P-DN \rightarrow D-UTN$) Transition Arc: this type of transition arc describes the influences of a process decision on the utility of the relevant disassembly object. A deep study on this transition arc reveals that for a fixed process, the possible disassembly objects that can be influenced by it have already been modeled in the Process Model: A **Process** has influences on several **Disassembly Objects** through the object property “*breaks*” and the object property “*creates*” (refer to [12] for detail description). Thus, the relationship between **Process** and **Disassembly Object** in the Process Model actually indicates the

influences of **Process_Decision_Node** on the **Disassembly_Object_Utility_Node**.

Thus, we don’t need to explicitly include that relationship in the Adaptive Disassembly Planning Information Model, rather the following semantic rule (shown in Table 2) has been added to transfer the relationship between the **Process** class and the **DisassemblyObject** class in the Process Model to the influence of the **Process_Decision_Node** on the **Disassembly_Object_Utility_Node** in the Adaptive Disassembly Planning Information Model:

Modeling of Type 5 ($D-UN \rightarrow D-UN$) Transition Arc: this type of transition arc describes the functional dependency between different disassembly objects. Refer to Figure 2, such information has already been modeled in the Uncertainty Model through “*functional Depends*” object property. Thus, the following semantic rule (shown in Table 3) has been added to the Adaptive Disassembly Planning Information Model to transfer the information in the Uncertainty Model for representing the causal relationship between two **Disassembly_Object_Function_Uncertainty_Nodes**.

R3->Model Integration: the implementation of this requirement has been already partially shown in the previous discussion. The Adaptive Disassembly Planning Information Model links the Process model, Uncertainty model and Degradation model through object property “*relatesTo*”. In detail, the integration is implemented in the following four places:

- (1) The **Process_Uncertainty_Node** class in the Adaptive Disassembly Planning Information Model links to the **Process** class in the Uncertainty Model, through which the relevant process related Conditional Probability Table (**Process Success Probability Table**) information can be retrieved.

Table 2: Semantic Rule R1 Definition

Semantic Rule: R1	
Antecedent (red line)	Consequent (blue line)
Process_Decision_Node(?x), Process_Utility_Node(?y), Process(?z), DisassemblyObject(?d), Disassembly_Object_Utility_Node(?u), influence(?x, ?y), relatesTo(?y, ?z), (<i>breaks</i> (?z, ?d) or <i>creates</i> (?z, ?d)), relatesTo(?d, ?u)	influence (?x, ?u)
Graphical Explanation	

Table 3: Semantic Rule R1 Definition

Semantic Rule: R2	
Antecedent (red line)	Consequent (blue line)
Disassembly_Object_Function_Uncertainty_Node (?x1), Disassembly_Object_Function_Uncertainty_Node (?x2), DisassemblyObject(?o1), DisassemblyObject(?o2), relatesTo(?x1, ?o1), relatesTo(?x2, ?o2), functionalDepends(?o1, ?o2)	influence (?x1, ?x2)
Graphical Explanation	

(2) The **Disassembly_Object_Function_Uncertainty_Node** class in the Adaptive Disassembly Planning Information Model links to the **Disassembly Object** class in the Uncertainty Model, through which the relevant function related Conditional Probability Table (**Function Failure Probability Table**) information can be retrieved.

(3) The **Process_Utility_Node** class in the Adaptive Disassembly Planning Information Model links to the **Process** class in the Process Model, through which regular and special process costs can be retrieved.

(4) The **Disassembly_Object_Utility_Node** class in the Adaptive Disassembly Planning Information Model links to the **Disassembly Object** class in the Process Model, through which the related recycle value, reuse value and discard cost can be retrieved.

Adaptive Disassembly Planning Application

This section discusses the detailed procedure for solving the adaptive disassembly planning problem. A high level view of the procedure is shown in Figure 3, which is an iterative process with two major involved sub-functions (indicated as green boxes in Figure 3):

F1->Component/Assembly Reuse Value Estimation. This function uses a fuzzy logic based approach for the component/assembly reuse value estimation. It takes the inputs from the human observation and further calculates the reuse value of the component or assembly and updates the Disassembly Decision Network accordingly.

F2->Disassembly Decision Making. This function carries out the Disassembly Decision Network based sequence optimization. It takes two types of information as inputs: (1) the component/

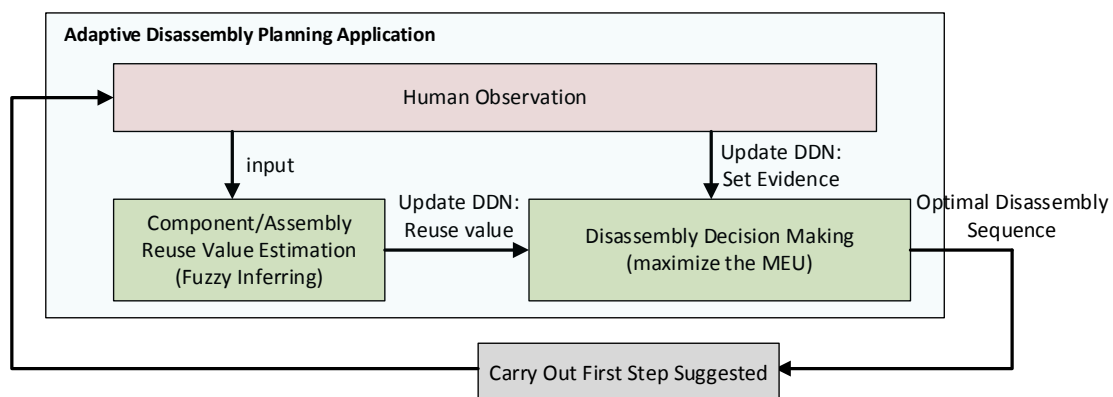


Figure 3: High Level View of the Adaptive Disassembly Planning Application

assembly reuse value (the output of F1) and (2) the human observation on whether or not a certain component/subassembly is functioning properly. The output will be an optimal disassembly sequence, based on the current available information.

After an optimal disassembly sequence is generated, the disassembly operator will carry out the first step in the suggested optimal disassembly sequence, which will yield a new disassembly state. New observation might be identified in the new disassembly state, which could affect both of the sub functions (F1 and F2). Thus, F1 and F2 will be re-evaluated based on the new observations and a new optimal disassembly sequence will be suggested. The whole process will iterate until the product is fully disassembled or the optimal disassembly plan becomes stable (remain same between iterations).

The following sections are organized as follows: two sub functions (F1 and F2) are discussed in detail in section 5.4.1 and section 5.4.2 first. Then, section 5.4.3 presents the complete application procedure in detail.

Component/Assembly Reuse Value Estimation: The goal of the first sub function is to estimate the component/assembly reuse value, which is an important information for constructing the Disassembly Decision Network. A concrete mathematical model to quantify this information is challenging and is very much case dependent. Thus, we use the idea of fuzzy inference, a technique that facilitates the modeling of a complex system without the knowledge of its mathematical description, for the reuse value estimation. In general, the fuzzy inference system consists of four modules as indicated in the Figure 4 below.

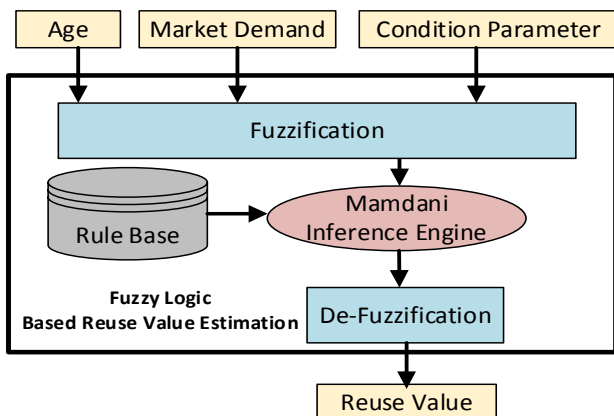


Figure 4: High Level View of the Fuzzy Inference System

Fuzzification module: Transforms the system inputs, which are crisp numbers, into fuzzy sets by applying the fuzzification functions. The system inputs are the critical variables identified in the Degradation Model and they are component/assembly age, market demand and a set of conditional parameters. It is assumed that these variables are sufficient to evaluate the component/assembly reuse value.

Each of the input variable is modelled as a linguistic variable, which is a composite data structure containing a set of fuzzy terms. Each fuzzy term further contains: (1) a linguistic value

(values are words in a natural or artificial language, e.g. Age = “Low”) which an input variable can take and (2) a membership function, which is used to quantify the degree of truth (0 to 1) of classifying a certain numerical value (e.g. Age = 2.5 years) into the linguistic value (e.g. Age = “Low”) the fuzzy term represents.

The following is the xml code for the “Age”, “Market Demand” and “Operation Noise” (condition parameter) linguistic variable.

```
<VariableVariableName="Age"LowerLimit="0"UpperLimit="5"VariableType="Input"><FuzzyTermName="Low"FunctionType="NormalMembershipFunction"Parameters="0,1.2"></FuzzyTerm>
```

```
<FuzzyTermName="Medium"FunctionType="NormalMembershipFunction"Parameters="2.5,1"></FuzzyTerm><FuzzyTermName="High"FunctionType="NormalMembershipFunction"Parameters="5,1.2"></FuzzyTerm>
```

```
</Variable>
```

```
<VariableVariableName="OperationNoise"LowerLimit="0"UpperLimit="50"VariableType="Input">
```

```
<FuzzyTermName="Normal"FunctionType="NormalMembershipFunction"Parameters="0,13"></FuzzyTerm>
```

```
<FuzzyTermName="Abnormal"FunctionType="NormalMembershipFunction"Parameters="50,13"></FuzzyTerm>
```

```
</Variable>
```

```
<VariableVariableName="MarketDemand"LowerLimit="0"UpperLimit="200"VariableType="Input"><FuzzyTermName="Low"FunctionType="NormalMembershipFunction"Parameters="0,50"></FuzzyTerm><FuzzyTermName="High"FunctionType="NormalMembershipFunction"Parameters="200,52"></FuzzyTerm>
```

```
</Variable>
```

Knowledge base: stores IF-THEN rules provided by experts. In this dissertation, four general rules related to the high reuse value and low reuse value are modeled:

Low Reuse Value Rule

R1: Age is High => Reuse Value will be Low

R2: Market Demand is Low => Reuse Value will be Low

R3: Condition Parameter is Worse => Reuse Value will be Low

High Reuse Value Rule

R4: Age is Low and Market Demand is High and Condition Parameter is Normal => Reuse Value will be High

Other customized rules can be added if needed and they can be acquired from the Degradation Information Model. An example is shown below:

Average Reuse Value Rule

Age is Medium and Market Demand is High and Condition Parameter is Normal => Reuse Value will be Average

Inference engine and Defuzzification: Fuzzy inference engine is the main decision making module in a fuzzy inference system. Its main operation is to convert the input fuzzy set into output fuzzy set through an inference process. Whereas, the defuzzification process transforms the fuzzy set obtained by the inference engine into a crisp value.

In this work, we use the popular Mamdani method for implementing the inference procedure and the details of the Mamdani method can be found here[14]. The defuzzification process is based on the idea of “Centroid of Area”, which returns the center of the area under the aggregated curve. If you think of the area as a plate of equal density, the centroid is the point along the x axis about which this shape would balance.

The sub function is implemented in Matlab and Figure 5 shows the implementation of the method for the reuse value estimation of subassembly ABCD.

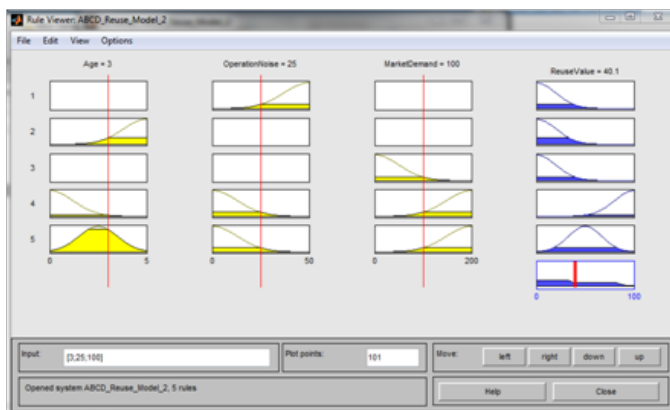


Figure 5: Fuzzy Influence Implementation in Matlab

As shown in Figure 5, the crisp input is [age= 3, Operation Noise=25, Market Demand=100]. Five fuzzy control rules are defined (four general and one customized, not shown in the Figure) and each will generate the fuzzy value of the output variable (ABCD reuse value). The five generated fuzzy values will then be aggregated and further again be translated into crisp value, the final inferred result is 40.1 (reuse value of subassembly ABCD).

Disassembly Decision Network based Disassembly Planning:

This section introduces a Disassembly Decision Network based adaptive disassembly planning approach, which integrates the Bayesian probability theory and the maximum expected utility (MEU) principle, for dynamically generating the optimal product disassembly sequence.

The determination of optimal disassembly sequence is to decide the value of each of the Process Decision Node (P-DN), which can yield a maximum disassembly object utility and a minimum the process utility (cost). If we annotate as one possible disassembly sequence, then can be expanded as follows:

$$d_i = pDN_i = \{pDN_{1,i}, pDN_{2,i}, pDN_{3,i} \dots pDN_{N,i}\}$$

Where is short for P-DN. The first subscript represents the index of the process decision node in the Disassembly Decision

Network, whereas the second subscript indicates the decision value (“carry out” or “do not carry out”) associated to that node.

Then, the expected utility (EU) of a disassembly plan is given by:

$$EU(d_i|e) = \sum_{p=1}^n \sum_{j=1}^2 \{P(pUN_{p,j})|e\} * pUTN_p(d_i, pUN_{p,j}) + \sum_{o=1}^k \sum_{j=1}^2 \{P(dUN_{o,j})|e\} * dUTN_o(d_i, dUTN_{o,j})$$

$pUN_{p,j}$ represents the Process Uncertainty Node in DDN. The first subscript represents the index of the process uncertainty node in the Disassembly Decision Network, whereas the second subscript indicates the uncertainty value (“fail” or “success”) associated to that node.

pUN_p represents the Process Utility Node in DDN. The subscript represents the index of the process associated to that node. can take different values depending on the arguments and .

$dUN_{o,j}$ represents the Disassembly Object Function Uncertainty Node in DDN. The first subscript represents the index of the Disassembly Object Function Uncertainty Node in the Disassembly Decision Network, whereas the second subscript indicates the uncertainty value (“function” or “not function”) associated to that node.

dUN_o represents the Disassembly Object Utility Node in DDN. The subscript represents the index of the disassembly object. can take different values depending on the arguments and .

e is set of evidence identified during the disassembly process.

The above equation describes the expected utility (EU) of a disassembly option given a set of evidences. It is an aggregation of two parts: (1) the expected utility of the disassembly process and (2) the expected utility of the disassembly object. Both of the parts are evaluated by calculating the summation of the relevant utilities, weighted over the probability values of the relevant uncertainty node.

In order to calculate the probability values like $P(dUN_{o,j})|e$ and $P(pUN_{p,j})|e$ Bayes rules are used here. In general, the basic task is to compute the posterior probability for a set of **query variables** (X) (in our case X is either $dUN_{o,j}$ or $pUN_{p,j}$), given some observed event—that is, some assignment of values to a set of **evidence variables** (e).

$$P(X|e) = \frac{P(X,e)}{P(e)} = \alpha P(X,e) = \alpha \sum_y P(X,e,y)$$

Y denotes the non-evidence, non-query variables Y_1, Y_2, \dots, Y_l (called the **hidden variables**) and α is the normalization constant.

Finally, the best decision D_* , given the probability distribution and the utility model is given by:

$$D_* = \max EU(d_i|e)$$

The above equation indicates that the optimal disassembly sequence, is a decision sequence which maximizes the $EU(d_i|e)$.

Complete Adaptive Disassembly Planning Procedure: The

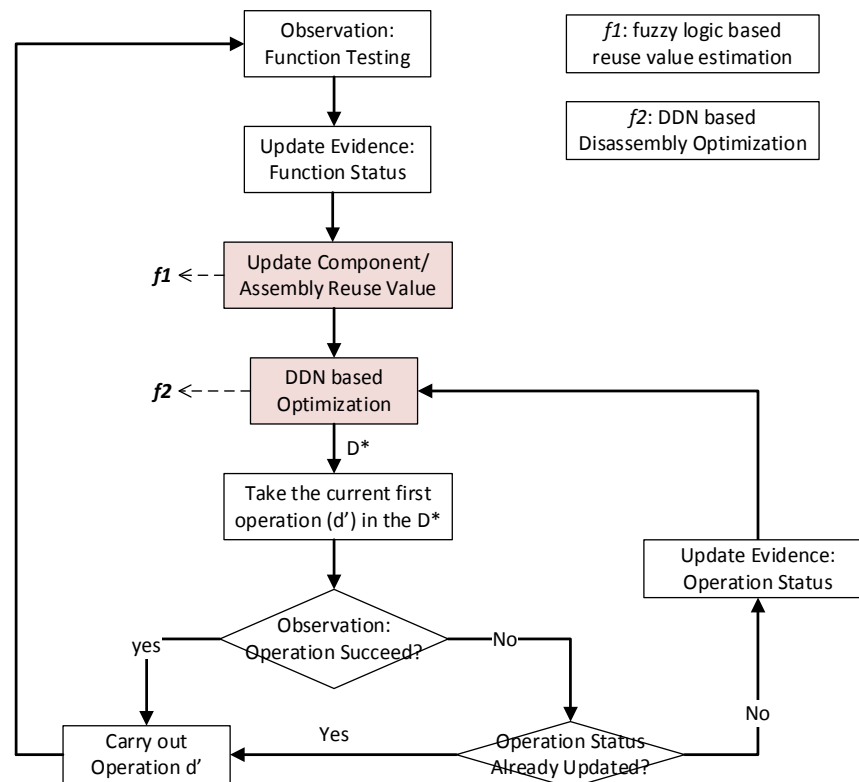


Figure 6: The Complete Adaptive Disassembly Planning Procedure

disassembly sequence (disassembly plan) generated above can only be considered “optimal” at the current disassembly state, in which only limited information or evidence can be identified. Carrying out one disassembly operation according to the plan puts forward the disassembly object to a new state with possibly more evidences revealed, which can change the “optimal” disassembly result generated by the DDN based Disassembly Planning function. Thus, the complete adaptive disassembly planning procedure is developed here (Figure 6) to iteratively generate optimal disassembly sequence.

When an EOL product is taken into the disassembly facility, function testing is going to be applied first, which will provide certain evidences on whether certain component/assembly is working properly or not. Notice that the function testing at this stage can only provide the functionality information about some of the components or assemblies. Whether or not the other components or assemblies are functioning properly is still an uncertainty to the disassembly operator. However, the probability of them to be functioning is updated based on the updated evidence. As an example, if the disassembly operator identifies that a fan assembly is functional and updates that information to the DDN as evidence, the probability of the motor to be functioning will be changed as follows:

$$P(\text{Motor} = \text{functioning}) \rightarrow P(\text{Motor}=\text{functioning} \mid \text{Fan Assembly} = \text{functioning})$$

Next step is to update the component/assembly reuse value in the DDN using the first sub function (f1: fuzzy logic based reuse value estimation), based on the identified age, market demand

and conditional parameter information.

The updated DDN will be sent to the second function (f2: DDN based disassembly optimization), which will generate an optimal disassembly sequence D^* , given the current available identified information.

The disassembly operator will take the first operation (d') suggested in the D^* to be the candidate disassembly operation at this stage. Another observation will be carried out to check whether d' can be successfully executed without problems. If d' can be successfully carried out, no updates in DDN are necessary and d' will be physically executed by the disassembly operator, which will yield a new disassembly state. Lastly, the process will re-route back to the beginning (function testing) and be applied to the new state.

On the other hand, if d' cannot be successfully carried out, an operation status update (status of d' =fail) is added to the DDN. With the new updated DDN, the optimization process (f2) is carried out again. Two possible scenarios can happen: (1) a new D^* will be generated to avoid d' and (2) same D^* which insisting on carrying out the original d' . The first scenario is straightforward, which indicates that the cost of executing d' , given the evidence the regular operation will fail, is not cost effective and should be avoided. On the other hand, the second scenario indicates that even though d' fails using the regular operation, some special operation (possibly with higher operation cost) should be applied, because the overall utility is still optimal compared to the other options. Lastly, same as the before, after carrying out one disassembly operation, either following the original d' or

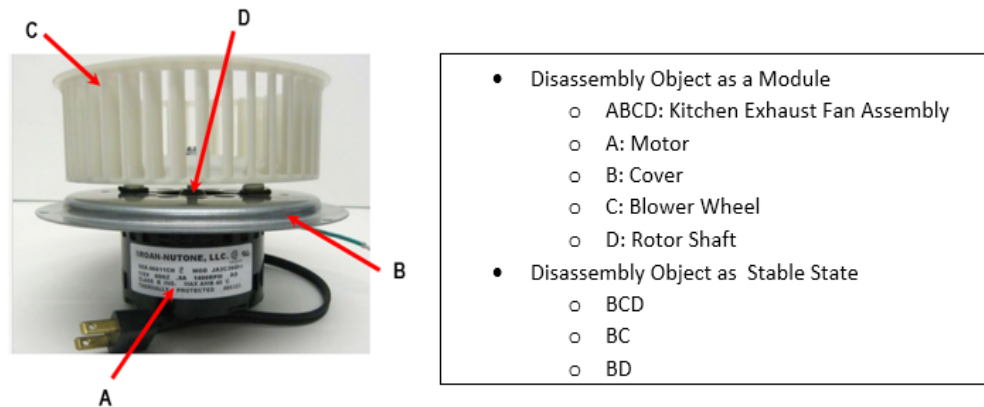


Figure 7: Kitchen Exhaust Fan Assembly

following the new d', a new disassembly state is being reached. The process will re-route back to the function testing step (beginning of the procedure), which will be applied to the new state.

5.5 Adaptive Disassembly Planner Application Verification

This section verifies the adaptive disassembly planning application using a kitchen exhaust fan assembly. We start with the description of the case study in section 4.5.1. The disassembly decision network for the case study is presented in section 4.5.2. Lastly, the detailed adaptive disassembly decision making process is verified in section 4.5.3.

Description of the Case Study: Figure 7 shows the pictorial presentation of the case study product, which contains four components (A->D) and one assembly (ABCD). Since all of them are associated with a designed function, they thus can have a reuse value after being disassembled. We call these type of disassembly objects module here.

On the other hand, three other subassemblies (BCD, BC and BD) exist only in the context of disassembly and they merely represent a stable state in the disassembly process and they don't have a designed function associated with them (i.e. they are not subassembly in the context of the assembly process). Thus, these type of disassembly objects don't have a reuse value. The utility related information is listed in Table 4 and table 5 below.

Table 4: Utility Information Regarding to the Disassembly Object

Disassembly Object	Reuse Value	Recycle Value	Discard Cost
ABCD	55	N/A	-10
BCD	N/A	N/A	-5
BC	N/A	N/A	-10
BD	N/A	N/A	-5
A	22	N/A	-15
B	10	3	N/A
C	15	2	N/A
D	10	4	N/A

Table 5: Utility Information Regarding to the Disassembly Process

Operation	Regular Operation Cost	Special Operation Cost
t1	-5	-10
t2	-8	-16
t3	-5	-20
t4	-10	-15
t5	-8	-15

Notice that in Table 2, there is no reuse value attached to for BC, BD and BCD because they do not have a designed function and they are only valid in the context of disassembly. Also, components B, C and D contain only homogeneous material, and thus they can always be recycled and no discard cost is assigned to them.

Another important information regarding this case study is the process model related to the product, which represents all the feasible disassembly sequences. It is shown in Figure 8 below, using the petri net as a pictorial notation.

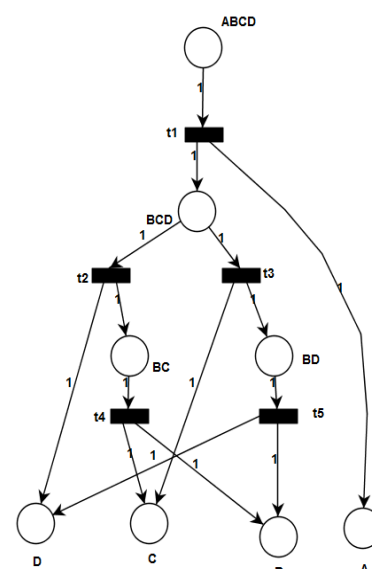


Figure 8: Feasible Disassembly Sequences of the Kitchen Exhaust Fan Assembly

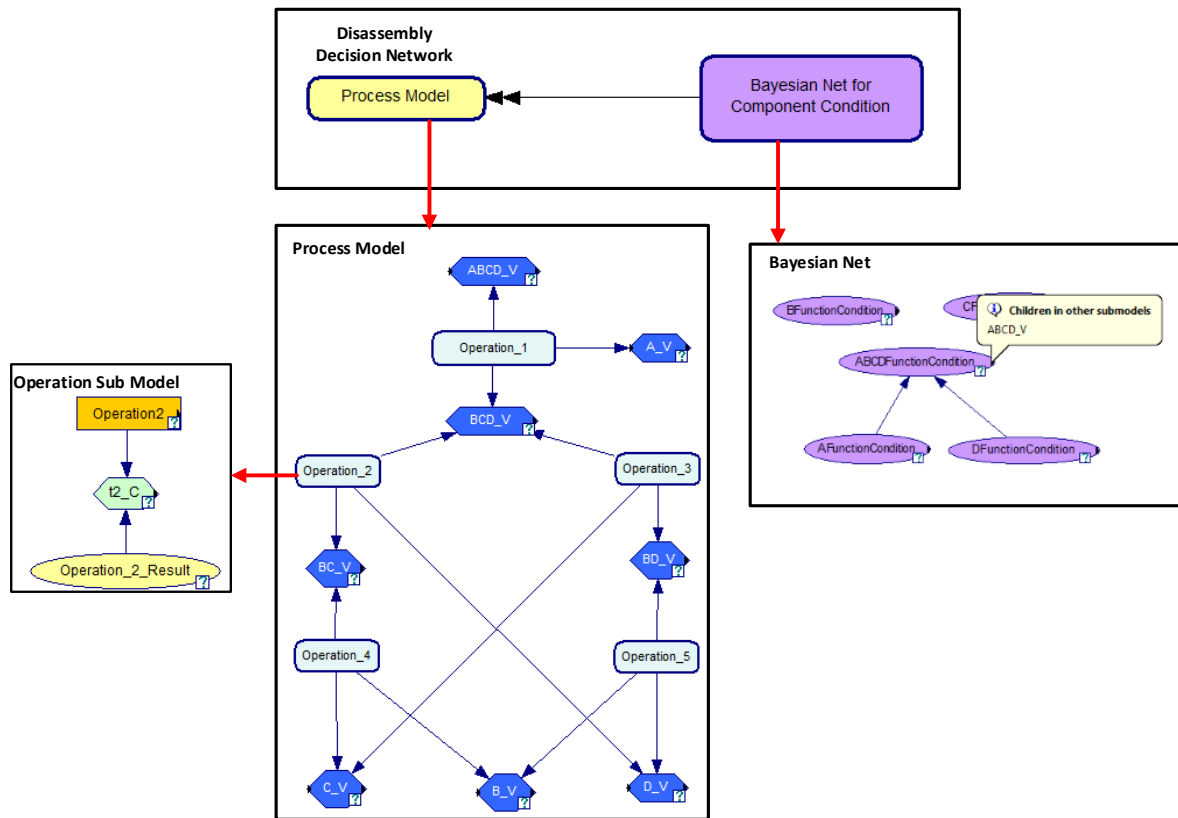


Figure 9: The Disassembly Decision Network of the Kitchen Exhaust Fan Assembly

Last but not least, the incoming product has two specific uncertainty issues, which are unknown to the disassembly operator in the beginning of the disassembly process:

- (1) Blower Wheel is not rotating (i.e. ABCD is not functioning).
- (2) Operation t3 cannot be executed in as a regular approach, some special process is needed.

Disassembly Decision Network for the Kitchen Exhaust Fan Assembly: Figure 9 shows the disassembly decision network for the kitchen exhaust fan assembly. In order to present it more concisely, the network has been partitioned into different sub models. On the top level, the disassembly decision network contains only two sub model: (1) process model and (2) Bayesian net model. The process model contains the information related to the disassembly object utility (node ABCD_V, A_V, etc.) and several operation sub models. Operation sub model is further an aggregation of the process utility, the process uncertainty and the process decision information. The Bayesian net sub model contains the disassembly object uncertainty information.

The model in the Figure 9 is a realization of the definition of DDN. Five types of nodes and five types of transition arcs are instantiated for the kitchen exhaust fan assembly, specifically they are:

Process Decision Node (P-DN): nodes “Operation1”, “Operation2”, etc.

Process Utility Node (P-UTN): nodes “t1_C”, “t2_C”, etc.

Disassembly Object Utility Node (D-UTN): nodes “ABCD_V”, “BCD_V”, “BD_V”, etc.

Process Uncertainty Node (P-UN): nodes “Operation_1_Result”, “Operation_2_Result”, etc.

Disassembly Object Function Uncertainty Node (D-UN): nodes “ABCDFunctionCondition”, “AFunctionCondition”, etc.

Transition Arc (Type 1: P-DN → P-UTN): e.g. arc pointing from node “Operation2” to node “t2_C”.

Transition Arc (Type 2: P-UN → P-UTN): e.g. arc pointing from node “Operation_2_Result” to node “t2_C”.

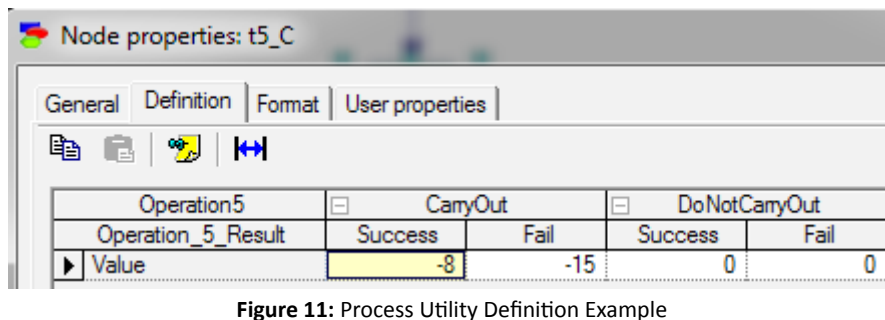
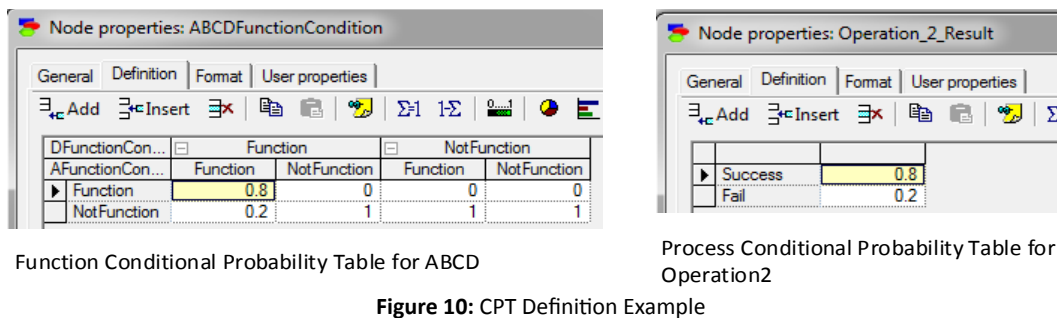
Transition Arc (Type 3: D-UN → D-UTN): e.g. arc pointing from node “ABCDFunctionCondition” to node “ABCD_V” (the arc is not shown in Figure 9).

Transition Arc (Type 4: P-DN → D-UTN): e.g. arc pointing from node “Operation1” to node “ABCD_V” (the arc is not shown in figure 9).

Transition Arc (Type 5: D-UN → D-UN): e.g. arc pointing from node “AFunctionCondition” to node “ABCDFunctionCondition”.

Also, conditional probability tables (CPT) are assigned to the relevant nodes. Figure 10 shows the user interface to input the CPT for both disassembly object function uncertainty node and process uncertainty node.

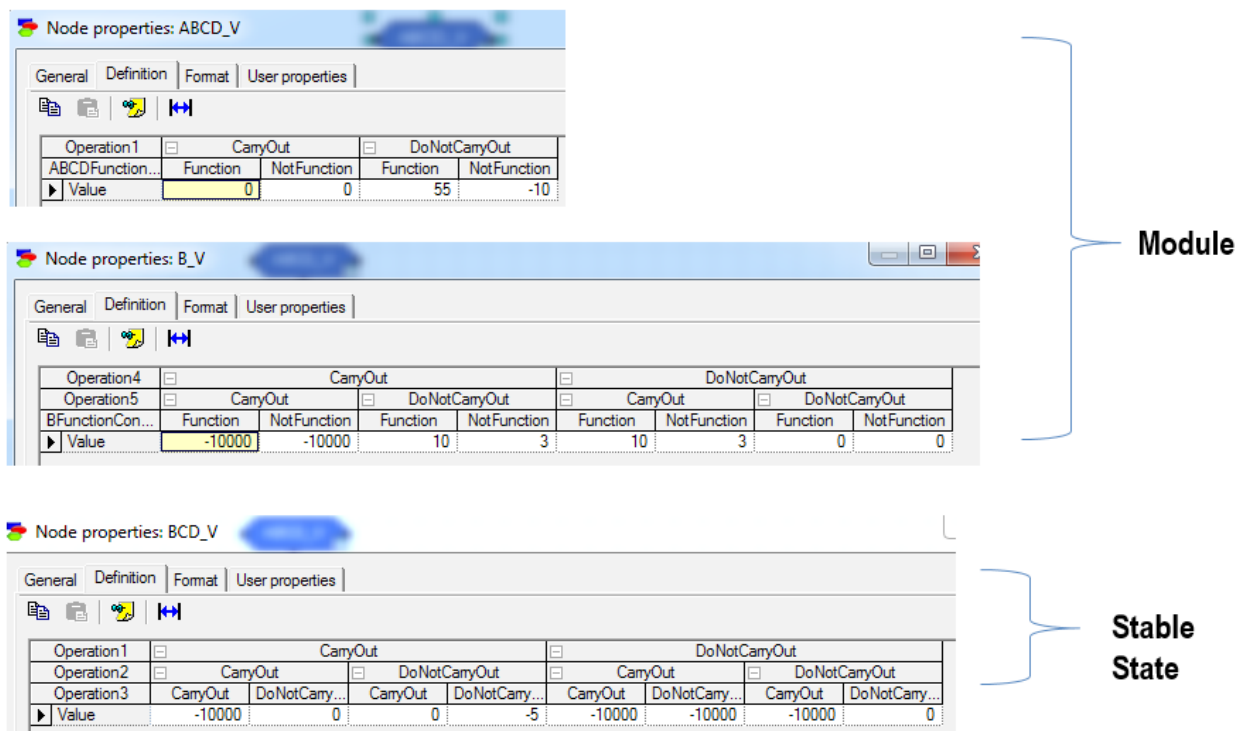
Lastly, the utility information (both for the disassembly object utility and the process utility) needs to be defined in the DDN. The process utility is relatively straightforward and it is based



on whether or not the operation is going to be carried out and whether or not the regular operation will success. Figure 11 shows an example of the utility definition for operation 5. As it is clear from Figure 11, the utility (cost) of operation 5 is zero under the condition that operation 5 is not carried out. On the other hand, if operation 5 is to be carried out, the utility (cost) will be either -8 (regular cost) or -15 (special cost), depending on whether or not operation 5 will be executed successfully without a problem.

The definition of disassembly object utility is classified into two categories: (1) the disassembly object is only a stable state and (2) the disassembly object is a module.

If the disassembly object represents a stable state, like the case for the subassembly BCD (Figure 8), the only variables influencing the utility are the operation decision nodes pointing to it. If we have 3 influencing operation decision nodes and each of which can take two decision values (“carry out” or “do not carry out”), we can have $2^3=8$ possible combinations. Each of the



combination will be assigned a utility value (either discard cost or recycling value). Some combination is realistically impossible, such as carrying out operation 1, 2 and 3 in the example in Figure 12. The utility of such cases will be set to a maximized negative number (-10000), which insures that it will not be selected as the optimal disassembly path. Some of the combination will yield a zero utility, which means the subassembly is further disassembled into smaller components and thus there is no utility (revenue or cost) associated with that.

Similar mechanism applies to the module type disassembly object, with only one extension: the function uncertainty node has an effect on the utility. If the module is functioning, reuse value could be applied to the utility value, otherwise the discard cost or recycling value will be applied.

Adaptive Sequence Generation for the Kitchen Exhaust Fan Assembly: The section verifies the adaptive disassembly sequence generation application using the kitchen exhaust fan assembly. The user interfaces of the developed adaptive

disassembly planning application are shown in Figure 13 below. Running the application using the kitchen fan assembly by

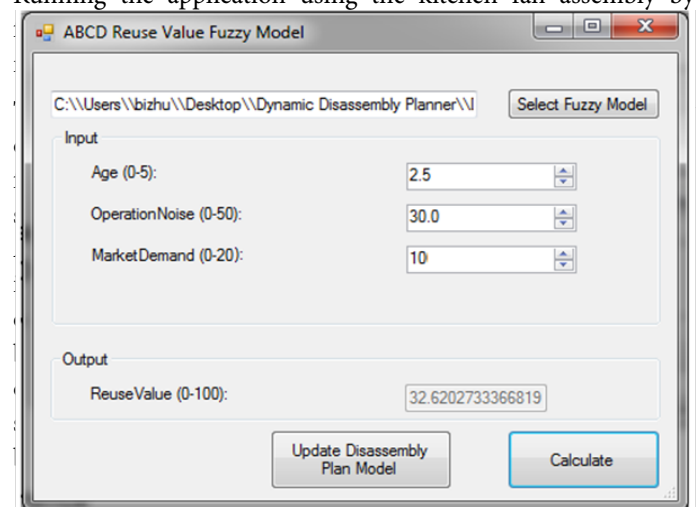
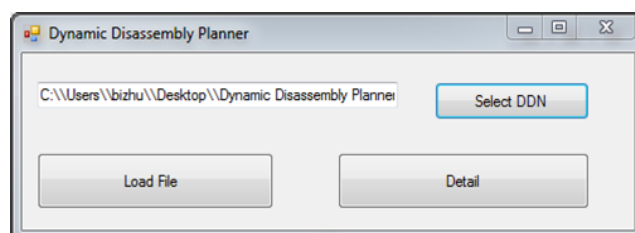
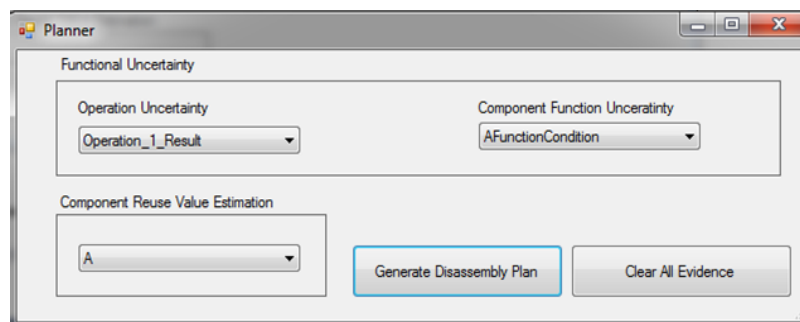


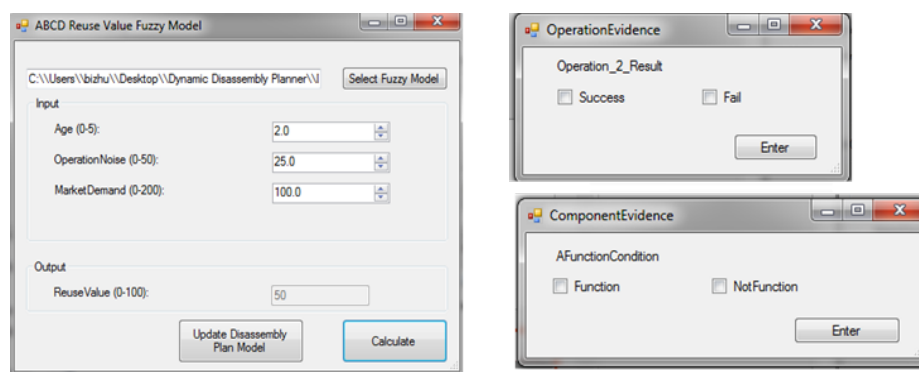
Figure 14: Reuse Value estimation for ABCD when Considering Degradation



Loading Disassembly Decision Network



Detail User Interface for uncertainty handling and Plan generation

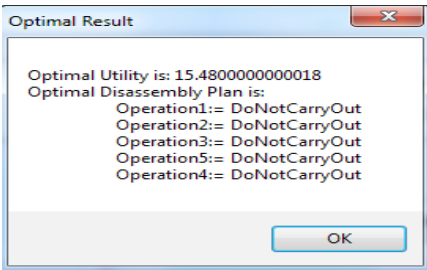
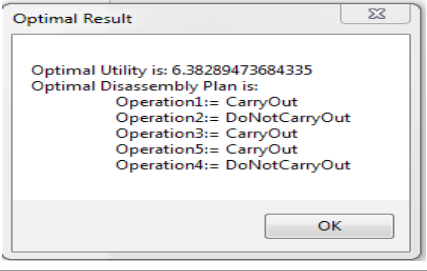
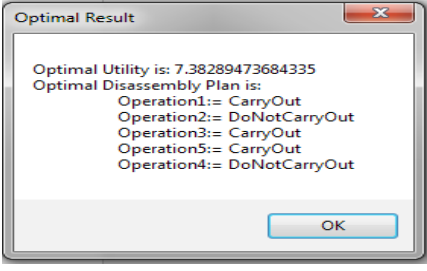
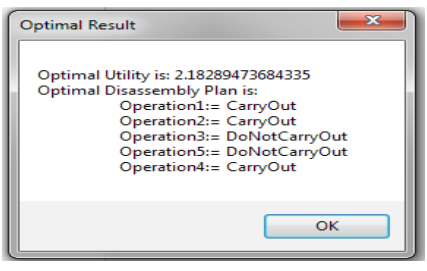


Evidence Updates

Fuzzy Logic Based Reuse Value Estimation and DDN Updates

Figure 13: User Interfaces for the Adaptive Disassembly Planning Application

Table 6: Adaptive Disassembly Plan for the Kitchen Fan Assembly

Stage	D*	Explanation
Initial Stage (No observation)		Do not Carry out any disassembly operation, retain the assembly ABCD, which will yield an optimal expected utility 15.48.
Stage 1: Function testing-> ABCD is not functioning		After function testing, the evident that ABCD is not functioning is updated to the DDN, a new D* is generated, which indicates to carry out operation 1, operation 3 and operation 5. This plan will in the end retrieve component A, B, C and D with a possible expected utility 6.38
Stage 2: Take the current operation in D* (Op1) as candidate and check whether it can be executed successfully -> Op1 can be executed successfully		Since Op1 can be executed successfully, the generated D* will remain same. However, the expected utility is increasing from 6.38 to 7.38 due to the new evidence.
Stage 2: Take the current operation in D* (Op3) as candidate and check whether it can be executed successfully -> Op3 can't be executed successfully		The evidence that Op3 can't be executed successfully is updated to the DDN, a new D* is generated. It suggests to carry out operation 1, followed by operation 2 and operation 4, which will avoid the failed operation Op3. The expected utility is 2.18 in this stage.
Both of the two uncertainties have been identified at this point, thus the plan from stage 2 will be the final plan (No change will happen to D* from this stage)		

Conclusion and Future Work

This paper presents Disassembly Planning Information Model, which constitutes a layered information framework designed for multiple applications in the domain of EOL product disassembly planning. DIM is hierarchically structured by layers, which divide the associated Information Models into different levels of abstraction, and thus, separate the general knowledge from the specific knowledge about particular domains and applications. A set of sub models is thus developed and classified into three different layers named the abstract level, the domain level and

the application level. The developed DIM is applied to the adaptive disassembly planning problem to validate the usability and reusability performance regarding the DIM.

While this work has demonstrated the utilization potentials of applying the DIM in the domain of disassembly planning, many opportunities for extending the scope of this thesis remain. This includes: (1) promoting the DIM into a reference model, (2) applying more disassembly planning related applications to the DIM and (3) integrating DIM with current IoT infrastructure.

References

1. GUNGOR A, GUPTA SM. Issues in environmentally conscious manufacturing and product recovery: a survey. *Computers & Industrial Engineering*. 1999; 36(4):811-853.
2. DONG J, ARNDT G. A review of current research on disassembly sequence generation and computer aided design for disassembly. *Proceedings of the Institution of Mechanical Engineers, Part B (Journal of Engineering Manufacture)*. 2003; 217:299-312.
3. WANG L, WANG XV, GAO L, VÁNCZA J. A cloud-based approach for WEEE remanufacturing. *CIRP Annals - Manufacturing Technology*. 2014; 63(1):409-412.
4. XIA K, GAO L, WANG L, LI W. A Semantic Information Services Framework for Sustainable WEEE Management Toward Cloud-Based Remanufacturing. *Journal of manufacturing science and engineering*. 2015; 137(6).
5. LEMAIGNAN S, SIADAT A, DANTAN J, SEMENENKO A. MASON: A Proposal For An Ontology Of Manufacturing Domain, Distributed Intelligent Systems: Collective Intelligence and Its Applications, 2006. DIS 2006. IEEE Workshop on 2006; 195-200.
6. CHANG X. *Ontology Development and Utilization in Product Design*, Virginia Polytechnic Institute and State University; 2008.
7. KUMAR PP. *Design process modeling: Towards an ontology of engineering design activities*, Clemson University; 2008.
8. KIM K, MANLEY DG, YANG H. Ontology-based assembly design and information sharing for collaborative product development. *Computer-Aided Design*. 2006; 38(12):1233-1250.
9. FOUFOU S, FENVES SJ, BOCK C, RACHURI S, SRIRAM RD. A core product model for PLM with an illustrative XML implementation. Geneve, Switzerland: Inderscience Enterprises Limited. 2005; 21-32.
10. BAYSAL MM, ROY U, SUDARSAN R, SRIRAM RD, LYONS K. The open assembly model for the exchange of assembly and tolerance information: Overview and example, 2004 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference; 2004 September 28 – 2004 October 2, American Society of Mechanical Engineers; 2004. p. 759-770.
11. FENG SC, KRAMER T, SRIRAM RD, LEE H, JOUNG CB, GHODOUS P. Disassembly process information model for remanufacturing. *Journal of Computing and Information Science in Engineering*. 2013; 13(3).
12. ZHU B. *An Information Model in the Domain of Disassembly Planning for Sustainable Manufacturing*, Syracuse University; 2016.
13. GRUBER TR. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*. 1995; 43(5–6):907-928.
14. Vukadinovic D. *Fuzzy logic: Applications, systems, and technologies*. Hauppauge. New York: Nova Science Publishers, Inc; 2013.