

Sustainable Manufacturing Analysis using an Activity Based Object Oriented Method

Yuriy Romaniw, Bert Bras and Tina Guldberg
Georgia Institute of Technology

Copyright © 2009 SAE International

ABSTRACT

This article begins by describing the need for a new method and tool for performing a sustainability assessment for manufacturing processes and systems. A brief literature survey is done to highlight the major existing methods and tools, their function, and their shortcomings. The article goes on to describe the general approach of the method before describing a computer aided tool that has been developed to implement the method. The article concludes with a walk through of a generic use case that describes where such a method would be useful and how such a tool would be implemented.

INTRODUCTION

In a modern world that is more aware of the scarcity of natural resources, it is becoming increasingly important to not only create and operate sustainable products but also be able to quantify and compare alternatives to best choose an optimal solution. A life cycle assessment (LCA) is one means of performing such a sustainability analysis. An LCA quantifies certain indicators throughout the entire life of a system. This type of assessment is known as a "cradle to grave" assessment. With respect to sustainability analysis, an LCA will focus on environmental impact indicators or resource consumption indicators. Such indicators can include, but are not limited to fuel consumption, raw material consumption, energy usage, emissions, and waste production. Current LCA tools generalize characteristics of a system during various stages of life that are based on a combination of empirical and

theoretical information. Existing LCA's, however, may be too general for parties that are interested in a specific stage of life. Of particular interest is the manufacturing stage of life. Other stages of life for a system, such as use phase and resource harvesting phase, have been well documented, but there is room for improvement when it comes to a sustainability analysis of the manufacturing stage of life. This article will focus on outlining a methodology and a tool that can help perform more detailed sustainability analysis of a manufactured part during the manufacturing phase of life.

MOTIVATION AND OBJECTIVE

Current LCA tools and methodologies are lacking in manufacturing phase information. To many industries that produce and sell products the manufacturing phase of life is the keystone to their operations. Industries, such as The Boeing Company may be able to perform generalized LCA's on how efficient their planes are during use phase, but Boeing is the primary operator of their products in the use phase. Boeing is not primarily involved in the design and manufacture phases of life for their products. Since the manufacturing phase of a products life defines such a large portion of this industries' operations, it is critical to have a way to perform a sustainability analysis on what it costs to manufacture a product, be it a plane, a car, or any other good.

The objective of this paper is to outline a methodology that can meet certain standards in helping carry out a more complete sustainability analysis of a manufactured part during the manufacturing phase of life. The method describes a model that is used to

perform such an analysis. This model must meet certain criteria:

1. The model must be flexible so that it can represent multiple different manufacturing processes
2. The model must be able to not only represent a system, but also execute simulations of that system
3. The model must be easy to use and modify, so that a user can readily define a system and easily interpret results

To meet the above requirements, a computer aided, object oriented, activity based costing model will be developed. An object oriented approach uses patterns of common elements to define classes that each have similar attributes and behaviors. Once properties have been defined, they do not need to be redefined every time a user creates another model. The user need only create instances of certain classes of elements, having those instances inherit properties and behaviors according to their base classifier. A computer aided approach will help with visualization and bookkeeping should the models get complex. An activity based costing (ABC) approach provides an underlying structure as to how the analysis would be carried out.

APPROACH

The model will be created in software called SysML (Systems Modeling Language). SysML is a plug-in for MagicDraw UML (Universal Modeling Language) and uses a graphical object oriented method to generating system models. To execute a simulation of a model, a SysML plug-in called ParaMagic, developed by InterCAX, which extracts data from the SysML model and performs mathematical computations in Wolfram Mathematica before updating the model with results. SysML is by no means the only software capable of creating such a model, it is simply the software tool used in this particular representation of the activity based, object oriented model (ABOOM) approach to sustainability assessments. Any software with similar capabilities should work. Additional details about the capabilities of SysML will be discussed later.

LITERATURE REVIEW

Nien-Lin Hsueh, et. al. [12] indicated that an object oriented approach is appropriate for constructing a model where repeatable patterns can be identified. Such repeatable patterns exist in manufacturing systems and are utilized by software, such as Computer Aided Manufacturing (CAM), Computer Resource Planning (CRP-II), and even more detailed Enterprise Resource Planning (ERP) software that broadens the scope of what is considered in the manufacturing process. A concept called Group Technology seeks out such similarities in order to help organize manufacturing system architecture. Group Technology lends itself greatly to object oriented modeling, because "the similarity in the characteristics of similar parts suggests that benefits can be obtained by classifying and coding

these parts into families." [13] Zhou Xiaoming and Du Pingan [21] outline an object-oriented approach to assembly sequence planning, showing that processes and not just parts can be commonly defined and classified. Hossam A. Gabbar [9] used an object-oriented approach to generate a design model for sustainable energy production and tracking of energy supply through a system. Gabbar created various classes that characterized energy, energy production methods, and other elements by generalizing common attributes amongst those elements. Freidenenthal [8] indicates the benefits of using an object-oriented approach to systems engineering by showing that an element does not change, even when viewed from different viewpoints. He argues that an element can be created as an object, and various designers can modify that object's attributes in order to describe that object from various perspectives in the design stage.

There have been multiple models that strive to assess an environmental impact or compute an ecological impact indicator of an element of interest. Software, such as EcoIndicator '99 and SimaPro, assign attributes to parts and use these attributes to compute a value that indicates the environmental impact of that part. Several companies, such as General Electric, have their own online tools that can give a quick and dirty assessment of impacts based on selection of appliances. Environmental management through activity based costing utilizes patterns to model interaction between three element classes; Cost Objects, Activities, and Resources Pineda-Henson and Culaba [15] analyze the sustainability of a semiconductor manufacturing process by integrating a traditional life cycle assessment with a decision making process. Combining traditional life cycle assessment metrics with the decision making criteria helped produce a tool that determines how sustainable a particular manufacturing process is. Emblemssvag and Bras [7] use an activity based approach to a product life cycle assessment.

Sustainability assessments tend to use ISO 14000 codes and regulations. Bennett and James outline a detailed list of many physical values and how they are used to compute an indicator. Pineda-Henson and Culaba as well as Emblemssvag and Bras [7,15] refer to ISO 14000 regulations, but also refine the list of physical quantities to reflect their particular approach. This model outlined in this article will not have a comprehensive list of physical quantities that lead to an indicator. As an initial proof of concept, this article will only look at emissions of carbon dioxide, NOx, and SOx, as well as quantify how much fuel and resources were consumed during the manufacture of a hypothetical part. The model will ultimately be expanded to include more physical quantities, as well as a means to compute an indicator. Since the model presented uses an object oriented approach, adding detail, such as additional physical quantities, is straightforward. The new value types would be created in the appropriate element, and the appropriate behaviors need only be defined once.

SYSML TERMINOLOGY AND NOMENCLATURE

Though SysML is a very versatile and comprehensive software, this model will only use a limited number of features. This may change as the model is expanded and reviewed.

The fundamental element that will be used is called a "block." A block can be used to represent any physical element or abstract idea, depending on the model. In this case, a block will be used to define objects and manufacturing operations. A block can contain many attributes, but the three primary attributes will be "references," "values," and "constraints." References are elements whose information is referenced by a parent block, but are not contained within a block. Reference are themselves blocks. Values are numerical quantities that are characterized by "value types." Value types defined the dimension and unit of a particular quantity. Constraints are similar to blocks, but represent mathematical relationships. Constraints have parameters that are related through a constraint equation. These parameters are value properties of a block. Blocks can also have many kinds of behaviors. This model will focus only on parametric behaviors that show how values amongst different blocks relate through mathematical constraints. Additional behaviors may be defined later as the model is expanded and reviewed.

"Instances" are, as the name suggests, specific examples of a block. A block is used to classify elements, while instances create unique examples of that class. Instances are what will store specific numerical quantities for values and they will represent unique elements in the model. A user would primarily be manipulating instances in the model, having had the blocks' attributes and behaviors predefined.

When elements appear graphically, their metaclass will appear in guillemets. For instance, when a block is represented, <<block>> will appear at the top of the elements. Similarly, value type will appear as <<value type>>, constraints as <<constraints>>, etc.

Camel case will be used as the primary writing style. This will assist plug-ins parse through the model, and will help maintain clarity in naming. The name of a class will begin with a capital or uppercase letter. For instance, let us say there is an element class called "manufacturing machine." This would appear as ManufacturingMachine in camel case. Names of attributes in a block will begin with a lowercase letter. Let us say a block owns an element called a drill press that is classified as a manufacturing machine. The words drill press would appear as drillPress. The combined attribute will appear as drillPress:ManufacturingMachine, which can be read as "an element called a drill press that is of class manufacturing machine." Instances will appear with an underline. For example, a Ryobi 12 inch drill press that is an instance of a Manufacturing Machine would appear as Ryobi12inDrillPress:ManufacturingMachine, which can be read as "an instance of the class manufacturing machine that is called Ryobi 12 in drill press." The name drillPress of the element appears only in the attributes of the block that contains the drill press, not in the instance

name. The instance, however, can be used to fill the drillPress attribute of the block.

When an element name is given beginning with a capital letter or in camel case, then this is meant to refer to a particular element of that designation within the model. If the element name is not capitalized or in camel case, it will be assumed it is a general usage of the word outside of the model. For instance: "ManufacturedPart" refers to an element within the model, while "manufactured part" is the general reference to a part that is manufactured, outside of the scope of the model.

ACTIVITY BASED COSTING METHOD

Traditional ABC defines an element that is called an "object." Objects are defined by or consume "activities," which in turn consume "resources." This can be seen graphically in Figure 1.

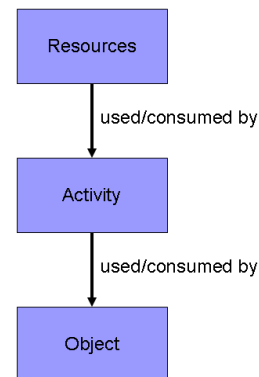


Figure 1: Traditional activity based costing breakdown

In this model, an object will be a manufactured part. Defining this part is a series of manufacturing operations that will function as activities. These operations will consume resources, which will be fuels, energy, materials, etc.

The traditional ABC structure is modified slightly in this model. The first key change is that activities can not only consume or use resources, but they can also produce waste. The second major change is that an object may contain other objects as well as activities, such as an assembly containing multiple parts and an assembly activity that joins them together. This allows for multi-scale system representation. The final significant change is that resources can, in some cases, be defined as objects that are produced by some sort of process described in terms of activities. An example of this would be a material resource like aluminum. Aluminum is refined (activity), and that refining process consumes resources.

Since this model represents a manufactured product, a naming convention will be used. The following elements will always be referred to with a capital or uppercase letter. An Operation is akin to an activity, and it can consume Resources and it can produce Waste. A ManufacturedPart is a fundamental manufactured element that can be defined completely in terms of Operations. A SubAssembly can contain multiple

ManufacturedParts and Operations that define how those parts are joined. Similarly, an Assembly contains SubAssemblies and Operations, while a Product contains SubAssemblies and Operations. Ultimately, a Product can be decomposed into a long list of Operations, each consuming Resources and producing Waste. This is shown graphically in Figure 2. Note that the words “Resources” and “Waste” are omitted, and are replaced with an “in” arrow, representing Resources, and an “out” arrow, representing Waste. The word Operation is abbreviated as “Op” and ManufacturedPart is abbreviated as “Part.”

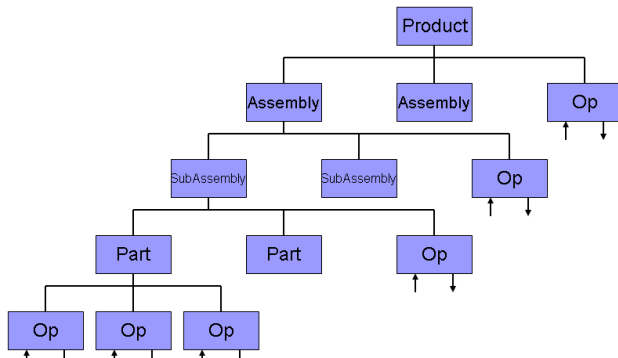


Figure 2: Structural breakdown of model elements

In addition to the above changes, there will be several other elements that are added to the model. The most common one will be an “actor.” Actors are contained in Operations and define at what rates Resources are being consumed and at what rate Waste and higher level objects such as Parts are being produced. Other minor changes will be either self-explanatory, or will be described when they appear in the model.

THE MODEL

CURRENT MODEL SCOPE

The model for this paper will focus on defining Parts and the Operations that describe them. The manufacturing phase of life for a ManufacturedPart will be the primary concern. Higher levels of the model will not be addressed in this paper and will be discussed further in the Future Work section. Several environmental impact quantities will be used, but this list is limited for now. This list includes consumable resources, such as fuels, raw materials, and energy, as well as waste such as carbon dioxide emissions, NOx emissions, and sulfur dioxide emissions.

MODEL STRUCTURE

There are three primary parts to this model. They are the Base Classifier Library (BCL), the Particular Element Library (PEL), and the Particular Instance Model (PIM). The BCL will be defined ahead of time within the tool and will be constant for all models. The PEL will be defined once by a particular user of the tool and will be referenced by the actual model. The PIM

will contain the final functional model and will be what the user will primarily be concerned with.

THE BASE CLASSIFIER LIBRARY

The BCL contains all of the generalized classes of elements that will be contained in the model. It will define all the attributes and behaviors of those classes, as well as any other classifier information that may be necessary. First, we define the Operation. All Operations will consume a quantity of fuel and a quantity of energy. They will all produce carbon dioxide, NOx, and sulfur dioxide emissions. This is shown graphically below.

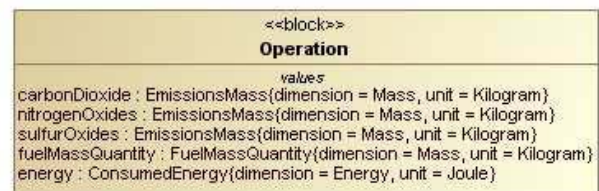


Figure 3: Definition of an Operation Class

These values are defined by value types that are shown in Figure 4.

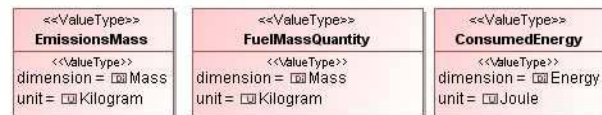


Figure 4: Value types used in Operation

This definition of Operation is general enough to capture some critical information, but it is too general for the purposes of this model. Therefore, three types of Operations will be defined, each having unique properties, but all being generalized by the definition of Operation in Figure 3.

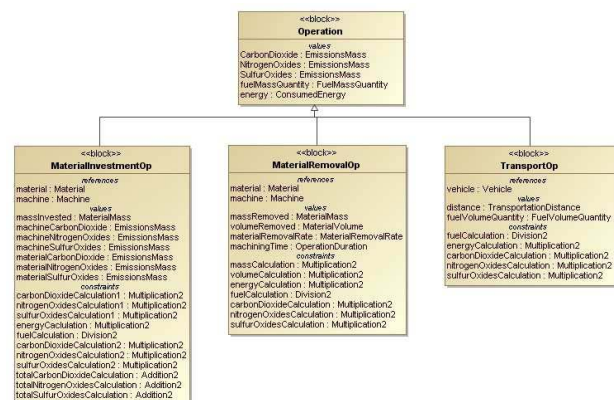


Figure 5: Specific Operation types

The three new Operation types that have been defined are material investment, material removal, and transportation operations. An example of a material investment operation is casting a ManufacturedPart.