# Object-Process Methodology as an Industry Enterprise Framework

Dov Dori[*], Ray Chow, Thomson David, Benjamin Koo, Christine Miyachi, Nathan Soderborg and Thomas Speller

*Engineering Systems Division*
*Massachusetts Institute of Technology*
*77 Massachusetts Avenue · Cambridge, MA 02139-4307 USA*

## Abstract

We present the principles of Object-Process Methodology (OPM) through its application to specification of systems from the automotive, IT and aerospace industries. Domain experts and senior system architects carried out the specifications and expressed the views in this paper. This effort was part of the coursework requirements in the Object-Process Methodology on-site/distance 2000 summer semester course within the Systems Design and Management (SDM) graduate executive program at MIT. The case studies demonstrate the expressive power of OPM as an intuitive yet formal enterprise framework modeling tool that can serve a common language among domain experts, system architects and software engineers.

## Keywords

Systems architecture, Object-Process Methodology, enterprise framework, pervasive computing

## Introduction

Object-Process Methodology (OPM)[1,2] is a holistic systems paradigm that extends the Object-Oriented (OO) paradigm and overcomes its major shortcomings by integrating system structure and behavior in a single integrated graphic and natural language model. Currently taught at MIT, Technion, Israel Institute of Technology and University of Rochester, OPM successfully tackles the task of development and lifecycle management of systems, products and projects. In industry it has experimentally been applied at such organizations as National Semiconductors, Ford Motor Company and Gemcor as a subcontractor for Lockheed in the NASA space shuttle project.

OPM is a significant extension of and a major departure from the OO approach. It incorporates the system static-structural and dynamic-procedural aspects into a single, unified model. Presented as a concise visual formalism by a set of Object-Process Diagrams (OPD set), it is automatically translated into a set of Object-Process Language (OPL) script, a subset of natural English.

At the basis of the OPM philosophy is the observation that to faithfully and naturally analyze and design systems in any domain, processes, like objects, should be considered as stand-alone “things” (entities) that are not necessarily encapsulated within objects. This detachment and de-coupling of processes from objects emphasizes the duality and complementarity of objects and processes, and opens the door for structure-behavior unification. At any point in time, objects exist with some structure and state. This is the static aspect of the system. Processes affect objects by changing their states. This is the dynamic aspect of the system. System complexity is managed through a number of graphical scaling options: zooming into and out of processes, unfolding and folding objects, and expressing or

[*] Contact author dori@mit.edu

suppressing object states. These mechanisms provide for selectively detailing a subset of things while still maintaining the high-level context of the details.
During the last five years, OPM has evolved from an analysis method into a comprehensive systems development methodology. Based on a unique unifying graphic language that is translated into Formal English (OPL) model, OPM encompasses the entire lifecycle of a system, a product or a project from concept and initiation through development to deployment and termination.
Currently experimented on at Ford and NASA, OPM has been successfully applied in a number of large-scale projects in USA, Germany and Israel. Application domains include (1) Business Process Reengineering of the technological knowledge base of a hi-tech metal cutting tool manufacturing firm, (2) designing a fully automated $800M semiconductor fabrication (FAB) facility, a re-engineered technological know-how of the world's fourth largest metal cutting tool manufacturer and (3) designing a wafer FAB cluster management system.
This paper presents the principles of OPM through the application of Object-Process Diagrams and Object-Process Language to specification of systems from the automotive, IT and aerospace industries. The system specifications presented in this paper were carried out by domain experts and senior system architects as part of their coursework in the Object-Process Methodology on-site/distance 2000 summer semester course[3] within the Systems Design and Management (SDM)[4] graduate program at MIT. We present and discuss the following industrial projects:

Mechanical System:
- Friction Stir Welding System by Gemcor/Lockheed for NASA

Electro-mechanical Systems:
- Automatic Fastening System for aircraft fuselage assembly by Gemcor for Israeli Aircraft Industry
- Anti-lock Braking System (ABS) by Ford Motor Company

Information Systems:
- Vehicle Dynamics Data Management at Ford Motor Company
- Reliability & Robustness System Management at Ford Motor Company

## Friction Stir Weld Joining Tool

NASA has expended considerable resources in the pursuit of system specification and reuse methodologies. As systems are becoming more complex, a prevalent problem in systems development is the amount of errors that accrue. These errors can cause catastrophic failure in the worst-case and intolerable schedule delays and cost overruns of pivotal projects. The inability to avoid introducing these errors in the first place, or tracking them down and successfully correcting them without introducing new ones in the process is primarily due to the fact that complex system specification is still carried out without proper analysis and design tools.

Gemcor is developing the Friction Stir Weld Joining Tool for the NASA/Lockheed-Martin external liquid oxygen and hydrogen fuel tanks of NASA's Space Shuttle. Gemcor Systems Corp. specializes in aerospace structure joining system development. The second system by Gemcor, described in the sequel, is a new member of a likewise new product family of Automatic Fastening Systems for Aero-structure Assembly for Israel Aircraft Industries (IAI). In the process of constructing the friction stir welding system we have little challenge in mastering individual technology elements. The complications in this project manifest themselves in the integration of construction, test and installation processes.

Figure 1 is a Catia-generated 3D drawings of the NASA space shuttle and the liquid oxygen (LO2) and hydrogen (LH2) fuel tanks. Figure 2 shows two frames from the Deneb-generated 3D-video simulation of the Friction Stir Welding process of the NASA liquid oxygen (LO2) and hydrogen (LH2) fuel tanks. This geometrical visualization can be useful for conceptual illustration. However, a more precise and symbolical representation is required to document the actual welding assembly processes. Gemcor used OPM to describe all the objects and processes involved in this welding activity.

Gemcor has provided the intent specifications to NASA/Lockheed-Martin in the form of a set of Object-Process Diagrams (OPDs) at an early stage of product development. The reason for selecting this approach was that the OPD notation is clear and concise to all external and internal customers responsible for the system design and use. Figure 3 is a System Diagram (SD) – the top-level OPD of the Friction Stir system. The OPD explicitly exposes the additional information that cannot be easily portrayed in the geometrical-based system drawings.

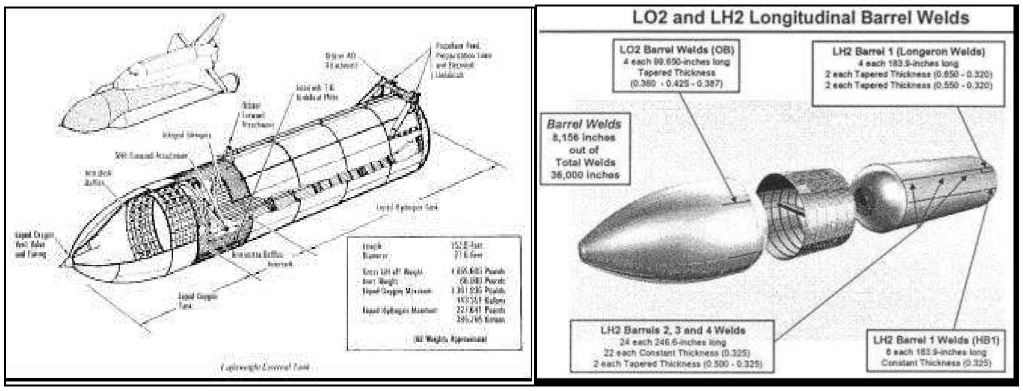
LO2 and LH2 Longitudinal Barrel Welds
LO2 Barrel Welds (OB)
4 each 99.650-inches long
Tapered Thickness
(0.380 - 0.425 - 0.387)
LH2 Barrel 1 (Longeron Welds)
4 each 183.9-inches long
2 each Tapered Thickness (0.650 - 0.320)
2 each Tapered Thickness (0.550 - 0.320)
Barrel Welds
8,156 inches
out of
Total Welds
36,000 inches
LH2 Barrels 2, 3 and 4 Welds
24 each 246.6-inches long
22 each Constant Thickness (0.325)
2 each Tapered Thickness (0.500 - 0.325)
LH2 Barrel 1 Welds (HB1)
6 each 183.9-inches long
Constant Thickness (0.325)

Figure 1. Catia 3D drawings of the NASA space shuttle and the liquid oxygen (LO2) and hydrogen (LH2) fuel tanks

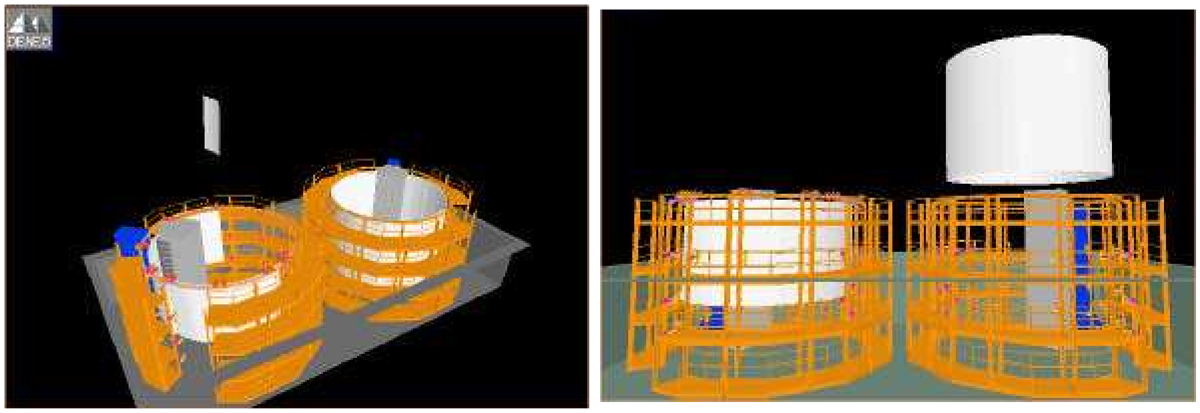

Figure 2. Deneb simulation of the Friction Stir Welding process of the NASA liquid oxygen (LO2) and hydrogen (LH2) fuel tanks

Gemcor intends to use OPM further for explanatory purposes for both Lockheed and NASA teams participating in the friction stir welding tool development. Gemcor will use OPM also for its alliance partner Pacific Aerospace and Electronics as a communication vehicle for commonly understanding the intent specifications as they are being formulated. A summary report will be provided to NASA and the public domain to help spread the word to various industry sectors, who are searching for a common communication and modeling vehicle for complex system development.

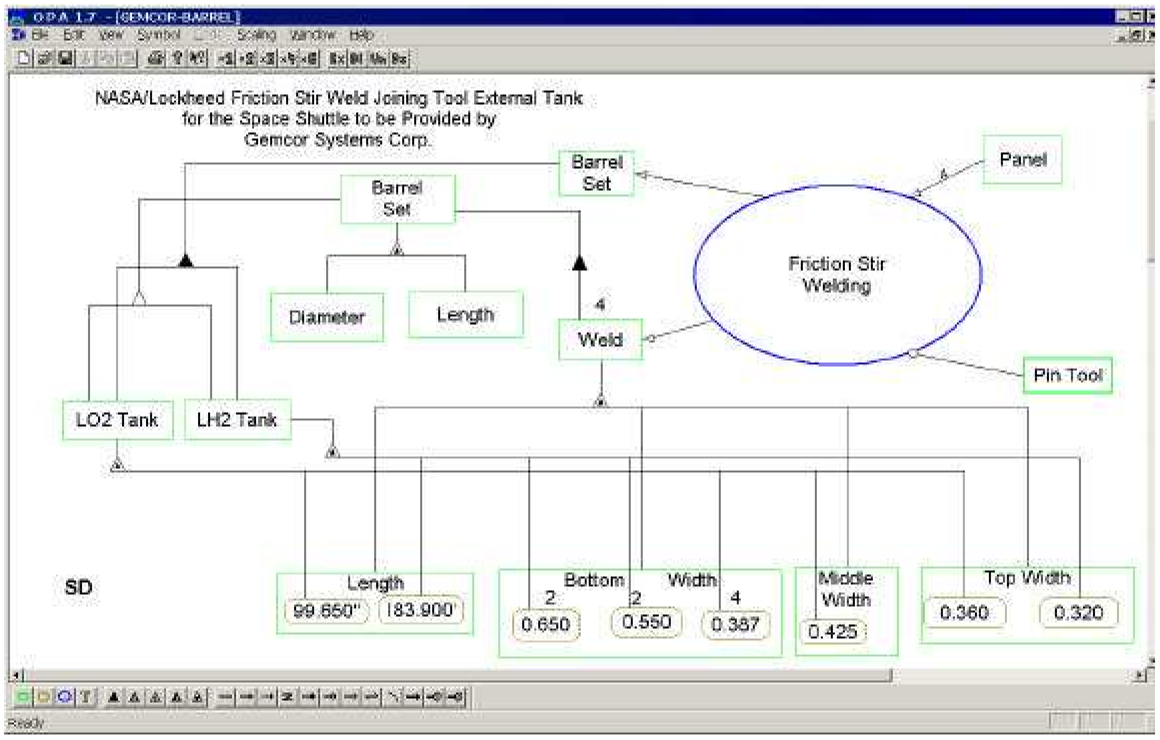

Figure 3. System Diagram (SD) – the top-level OPD of the Friction Stir Welding of the NASA liquid oxygen and hydrogen fuel tanks

## Automatic Fastening System

The Automatic Fastening System is more complex than the Friction Stir Welding system. Featuring an all-electric design, compared to the hydraulic past practice, it offers the advantages of 50% fewer parts, half the cost, half the time-to-market, is 25% faster and has greater precision in the motion axes. The greater speed provides faster assembly throughput and the higher quality translates into greater fatigue life and joint strength for the airframe. The application in this case is the automatic assembly of the entire fuselage of a new German Dornier Regional Jet. IAI is a risk-sharing partner with Dornier and will produce the fuselage in Israel. The system has 10 axes under CNC control with fiber optic multiplexed I/O network shared among all positioning and control sensors and actuators. Special high speed and precise closed loop change on the fly servo controls are used in the automatic fastening cycle, called the Drivmatic® Cycle.

The intent specification was written in OPM. The software code is a layered architecture and is quite complex. The OPD/OPL specification would require an extreme amount of detailed description of each layer with post processors written for each control software type. It may be less time and cost intensive to maintain the OPD/OPL at a higher level but sufficiently thorough to guide the programmers in adhering to the intent specifications.

Similarly to the Friction Stir Welding, the Automatic Fastening System is Catia/Deneb simulator generated model that provides a clear motion picture of the system to be delivered. It serves both as a design compliance tool and as a CNC positioning code generator via a COTS post processor for the CNC. Considerable time, cost and risk mitigation are saved with the simulator.

The Automatic Fastening System appeared initially to be only about twice as complex as the Friction Stir Welding System. However, after in-depth OPD/OPL development and counts structure hierarchy levels (zoom-ins and unfolding operations), the automatic fastening system is estimated to be approximately three times as complex which was a surprise to Gemcor. Figure 4 is a System Diagram of the Automatic Fastening System for passenger aircraft fuselage.

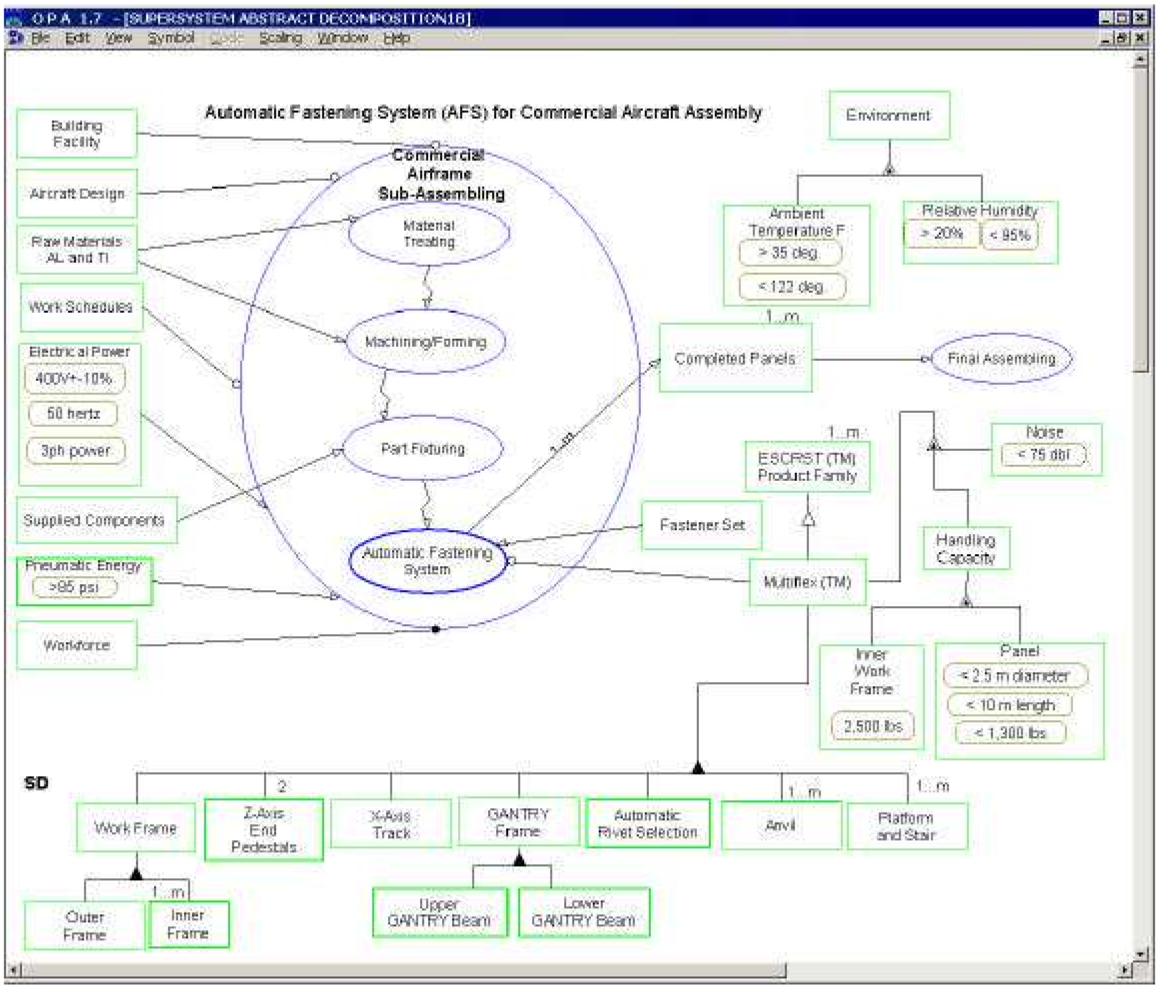

Figure 4. System Diagram (SD) of the Automatic Fastening System for passenger aircraft fuselage

## Applying OPM to the Anti-Lock Brake System (ABS) Design

Most of today's vehicles are equipped with an Anti-Lock Braking (ABS) system. ABS is a semi-control-feedback that integrates software and hardware functions. It automatically senses a "wheel lock" condition and applies the optimal brake pulse to release the locked wheel in order to avoid slippage as much as possible. Figure 5 shows two schemes of the hydraulic system in a car with ABS.

As vehicles are being transformed from pure mechanical systems to intelligent control systems, ABS is becoming a model for new automobile subsystem components. Throughout the history of ABS development, engineers have been suffering from lack of an adequate tool to effectively express the complexity of the software-hardware interaction. With OPM now becoming available to the engineering community, Ford engineers have explored OPM's capability to abstract and simplify design in the automobile engineering world.

**HYDRAULIC SYSTEM OPERATION**

As we now know, the hydraulic system operates in four modes: Non-anti-lock braking, pressure isolation during anti-lock braking, pressure modulation/decrease during anti-lock braking and pressure modulation/increase during anti-lock braking (Fig 3.). Each of these operational phases are described as the action occurs.

1. 4WAL EHCU
2. Combination Valve
3. Pressure Warning Switch
4. Master Cylinder
5. Booster
6. Brake Pedal Switch
7. Parking Brake Switch
8. 4WD Switch
9. Wheel Speed Sensors
10. Warning Lamps

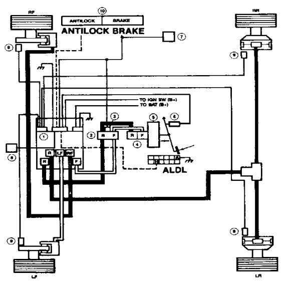

*Fig. 3: Hydraulic Operating System & Components of 4WAL*

**SYSTEM CONFIGURATION**

The General Motors Four-Wheel Anti-Lock (4WAL) brake system consists of a conventional braking system (front disc, rear drum), an Electro-Hydraulic Control Unit (EHCU) containing an internal Electronic Control Unit, hydraulic control unit assembly and electric pump, wheel speed sensors at each wheel, two warning lights on the instrument panel, a brake switch and all necessary wiring and connections. *See Fig. 1.*

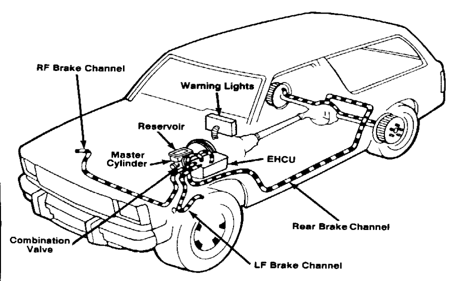

*Fig. 1: Typical 4WAL Component Locations on General Motors Vehicles*

Figure 5. Schemes of the hydraulic system in a car with ABS

This OPM modeling has focused on the internal functionality of the ABS brake system by showing through a hierarchical decomposition of the system how software and hardware interact. OPD clearly illustrates the context of how the ABS algorithm affects the surrounding hardware components. The OPM model provides for a better understanding of how the ABS algorithm determines a wheel lock condition and how it consequently applies a series of braking pulses to relieve this undesired condition.

This study has also demonstrated the capability of OPD and OPL as a design tool to systematically decompose a complex system to manageable chunks without losing critical information. The folding and unfolding of the objects provides the designer with a meaningful way of scaling up and down of the hardware design while maintaining clearly defined relationships at all levels of abstraction. Hardware features are captured through a characterization link that is uniquely defined in OPM and expressed in both graphics (OPD) and text (OPL).

Figure 6-left shows the conventional "hardware system decomposition diagram" while Figure 6-right shows the system interface diagram. While both diagram types are in common use in today's mechanical design processes, their success in expressing the complexity of the ABS software-hardware interaction has been rather limited.

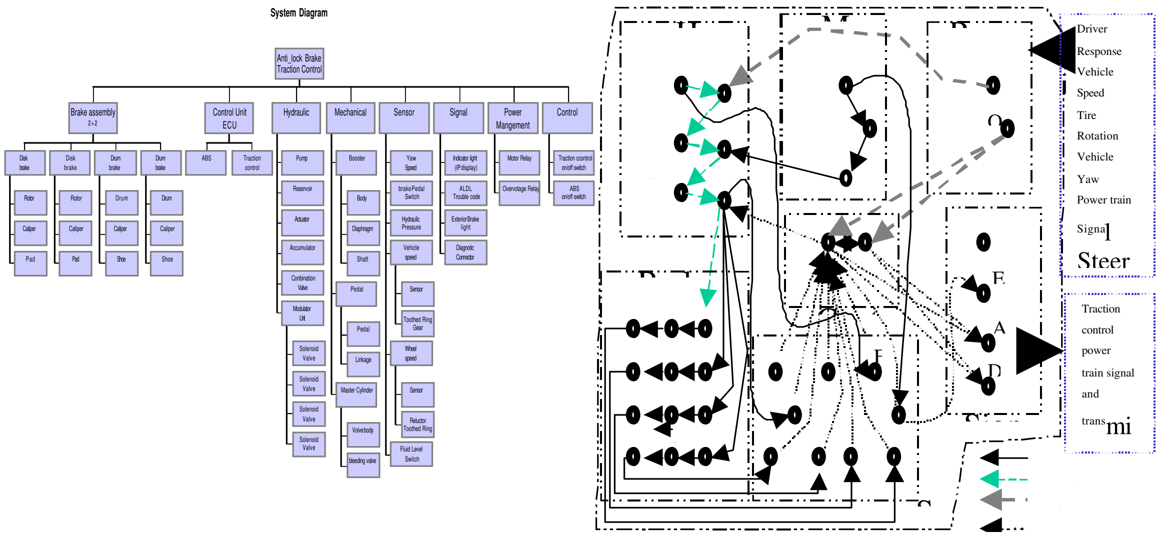

Figure 6. ABS hierarchy elements (left) and interfaces (right)

With a set of OPDs, and the corresponding OPL script specification, the structure and dynamics of the entire system at all the detail levels is captured. As the processes zoom in, the intimate interaction between the hardware and the software become clearly visible. As the ABS System Diagram (top level OPD) in Figure 7 shows, the skeleton of the system is clearly laid out to unveil its subsystem hardware components, its software processes and the interactions among them. The corresponding OPL paragraph in figure 6 likewise textually explicates the natural language OPL sentences corresponding the System Diagram.

The final detail texture will be fully disclosed as we continue this zoom into the system's processes. To avoid any unnecessary confusion, only necessary information needs to be exposed to the design engineer.

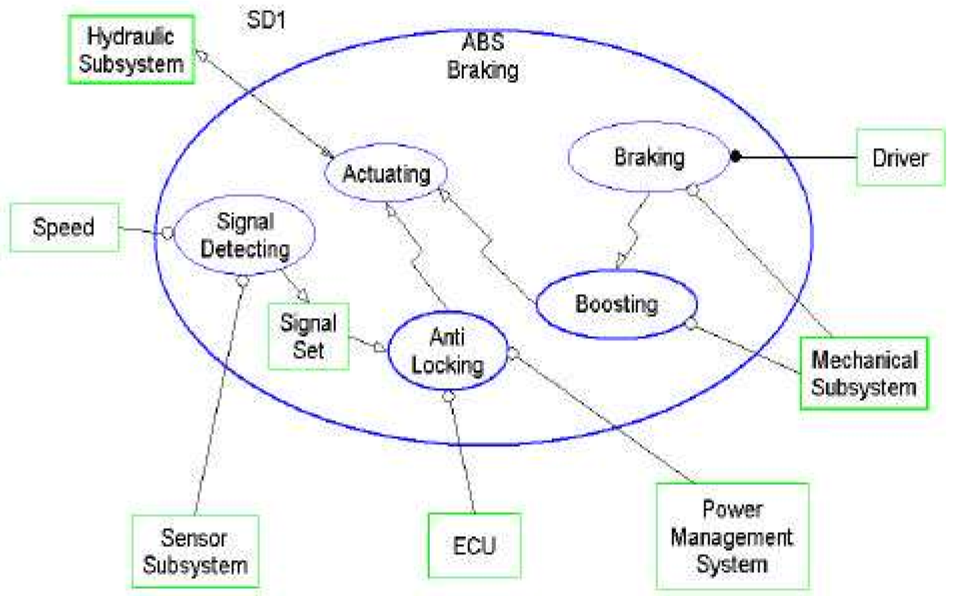

Figure 7. Top level OPD of the ABS

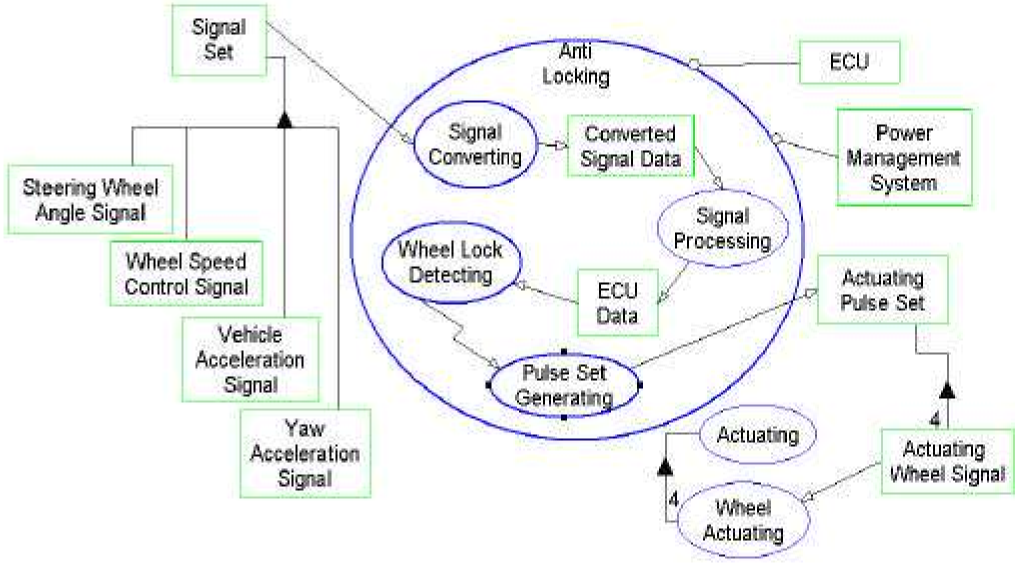

Figure 8. The OPD in Figure 7 detailed: Signal Set unfolded and Anti Locking zoomed

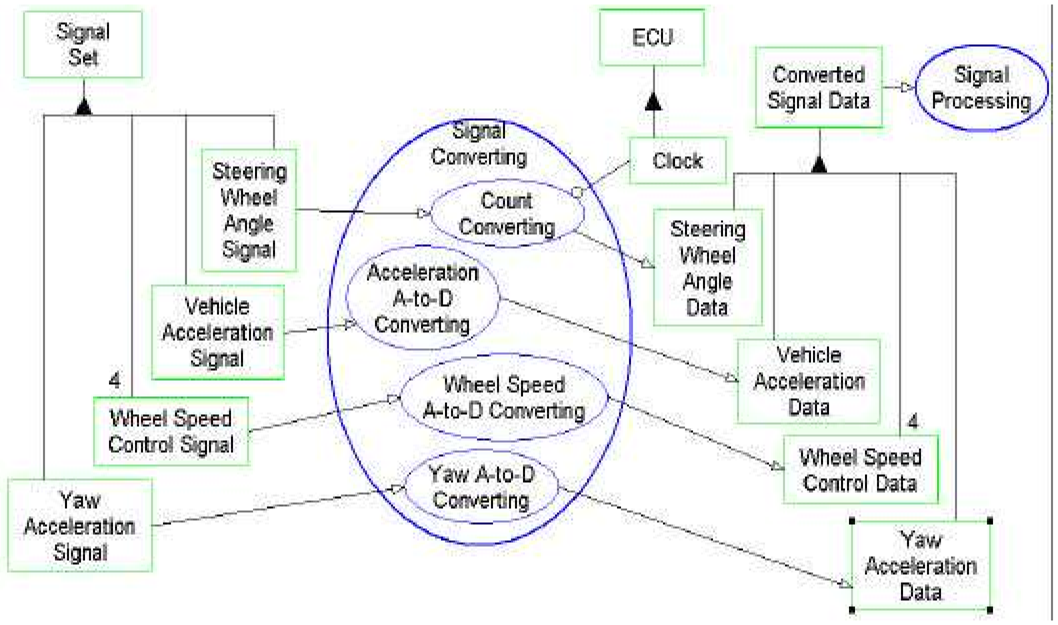
Signal Set
ECU
Converted Signal Data
Signal Processing
Signal Converting
Clock
Steering Wheel Angle Signal
Count Converting
Steering Wheel Angle Data
Acceleration A-to-D Converting
Vehicle Acceleration Signal
Vehicle Acceleration Data
4
Wheel Speed A-to-D Converting
Wheel Speed Control Signal
4
Wheel Speed Control Data
Yaw A-to-D Converting
Yaw Acceleration Signal
Yaw Acceleration Data

Figure 9. The OPD in Figure 8 detailed: Signal Converting zoomed, Signal Set and Converted Signal Data unfolded

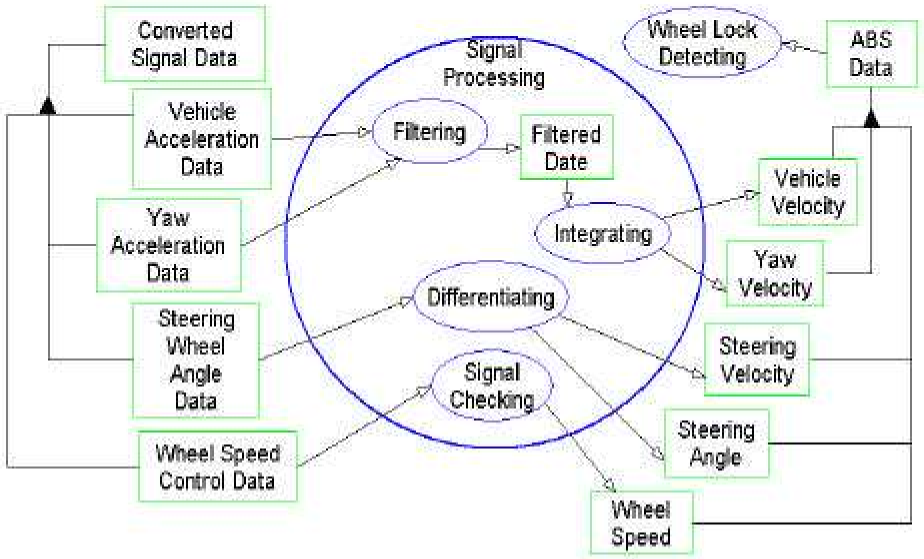

Figure 10. The OPD in Figure 9 detailed: Signal Processing zoomed, ABS Set Data unfolded

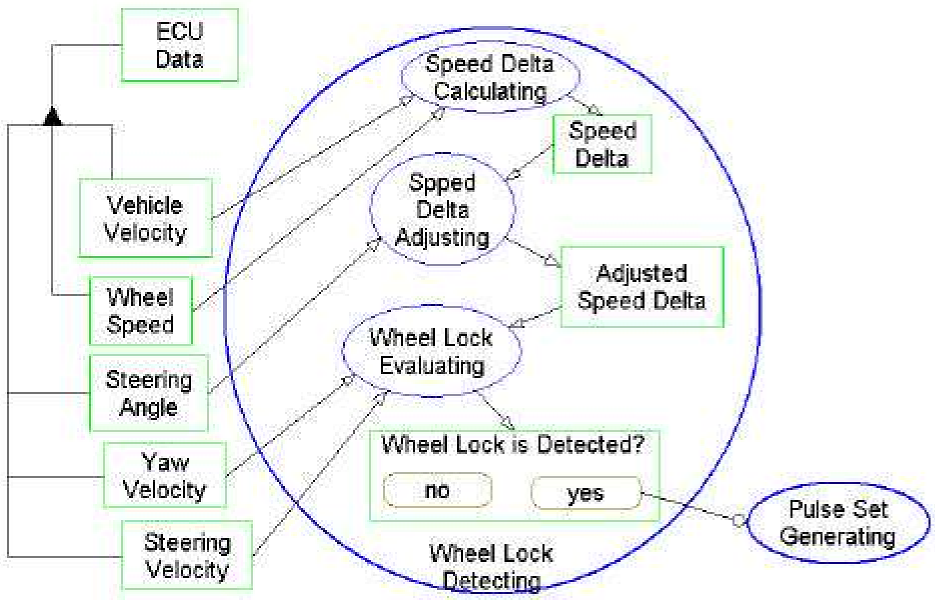

Figure 11. The Wheel Lock Detecting process of the OPD in Figure 8 zoomed, ECU Data unfolded

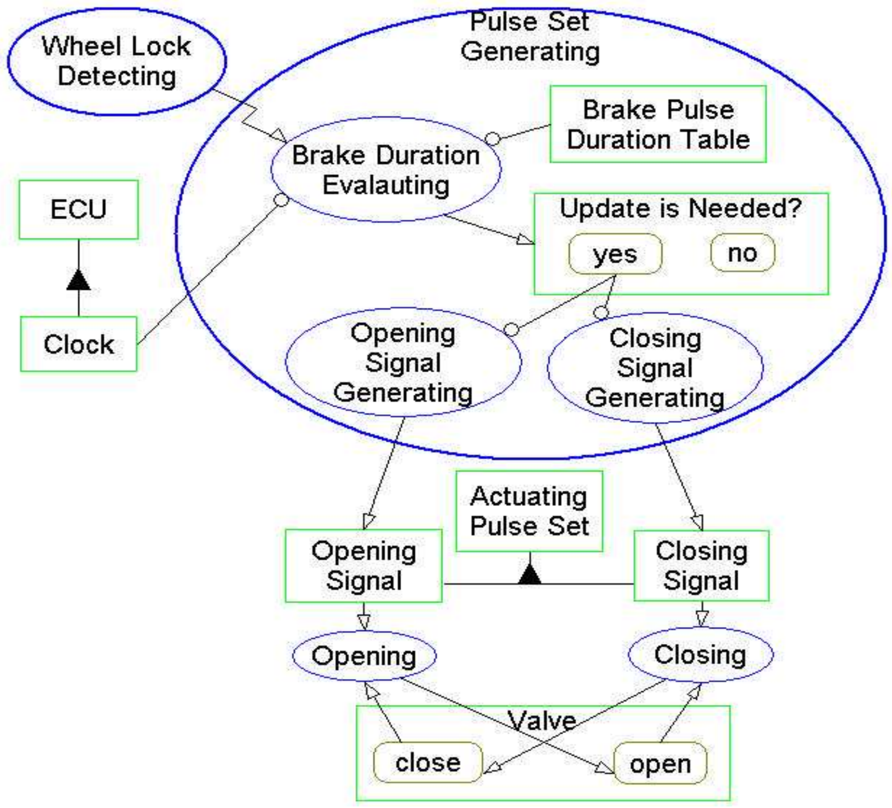

Figure 12. The Pulse Set Generating process of the OPD in Figure 8 zoomed, showing the opening and closing of the ABS valve

To summarize our experience in the ABS case study, OPM provides a new framework to capture the complexity of hardware and software interaction. Through OPL, it is possible to translate the process into a machine executable code. In addition, OPM can capture the dynamic behavior of the hardware attributes and software states in a single graphical language. It eases the development effort for evaluating the system reliability during the design stages. Simulation and testing protocols can be automatically generated though future extensions of OPM to reduce lengthy system verification efforts.

## Applying OPM to Vehicle Dynamics Data Management

Vehicle Dynamics Test Activity at Ford Motor Company performs vehicle dynamics testing and analysis of the data, generating reports for the requesting departments. The present form of data management is not efficient and lacks documentation. In this project an attempt was made to use Object Process Methodology (OPM) to design and develop the data management operation of the Vehicle Dynamics Test Activity.

The source of vehicle dynamics objective data comes from testing in three main areas – objective handling, ride, and laboratory measurements. The tests are initiated with a Test Request from various requesting departments. This operation needs to keep track of all the tested vehicles, the test requests received, and associated analyzed test results. Apart from keeping track of test data, an attempt was made to define the process of testing using OPM. The database should have the capability of performing functions such as:

- automatically uploading of data from the vehicle dynamics analysis software;
- allowing users to search and sort vehicles tested;
- comparing vehicles for various specified metrics;
- creating standard reports;
- archiving the raw data; and
- providing an automatic update to vehicle dynamics SDS (System Design Specification).

The database should be accessible from all locations of Ford Motor Company around the world. The proper management of the data will allow the department to achieve its goal to maintain leadership in the development of Vehicle Dynamics Test Methodologies and transferring these methodologies to other development community.

## Current Status in Data Management Process Modeling

The Vehicle Dynamics Test Data Management is a complex operation. The attempt to model the vehicle dynamics testing operation and a process to manage the data using Object Process Methodology (OPM) was very successful. Using Object Process Diagrams (OPDs) and Object Process Language (OPL), the operations were well represented without any ambiguity whatsoever, while the design implementations of major operations were complete.

## Top Level OPD and OPL

Requesting departments will submit a Test Request and Vehicle to conduct various vehicle dynamics tests to the Testing activity. Test Request will contain all the vehicle and test specifications and details of the tests to be performed. All the tests are conducted on a Test Track located in Dearborn, Florida, Arizona or Europe. Based on the Test Request the Technologists will conduct various vehicle dynamics tests to generate Vehicle Dynamics Test Data. The test data will be stored in a database. The data will be checked for validity before analyzing using custom software. The analysis will generate test reports and output metrics to be stored in database for later use. It will also generate a report, which may be stored in the office as a backup copy, and the requesting department gets the original report. Figure 13 shows the top-level System Diagram for the vehicle dynamics test management operation.

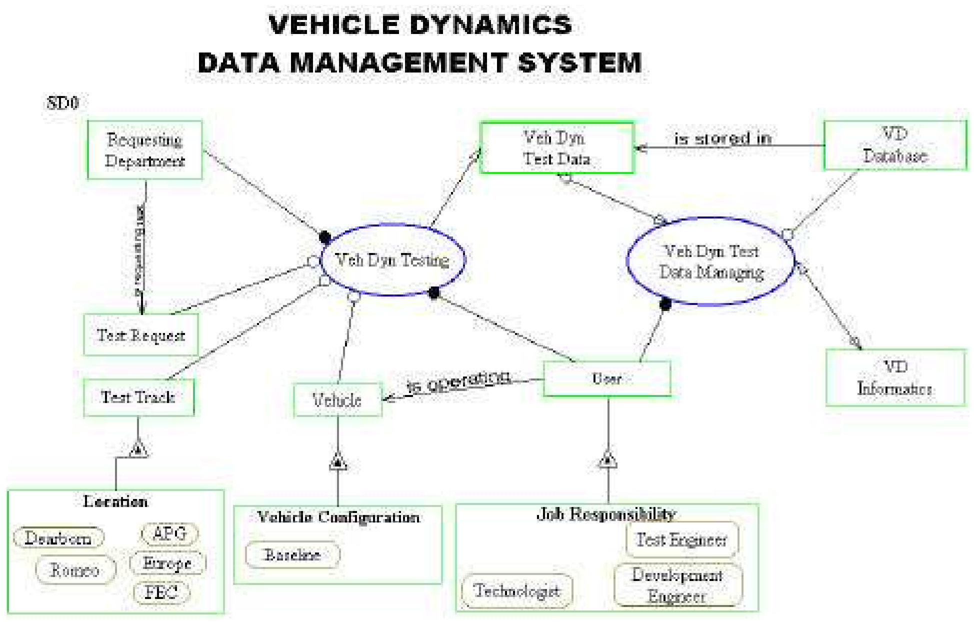

Figure 13. System Diagram of the Vehicle Dynamics Data Management System

The associated OPL for the operation is provided in Figure 14.

**Vehicle Dynamics Testing** requires **Vehicle, Test Request,** and **Test Track.**
**Vehicle** exhibits **Vehicle Configuration.**
**Vehicle Configuration** can be **Baseline.**
**Test Track** exhibits **Location.**
**Location** can be **Dearborn, Romeo, APG, FEC** and **Europe.**
**Requesting Department** and **User** handles **Vehicle Dynamics Testing.**
**Requesting Department** is requesting test by **Test Request.**
**Vehicle Dynamics Testing** yields **Veh Dyn Test Data.**
**Veh Dyn Test Data** is stored in **VD Database**.
**Veh Dyn Test Data Managing** requires **VD Database.**
**Veh Dyn Test Data Managing** affects **VD Informatics.**
**Veh Dyn Test Data Managing** processes **Veh Dyn Test Data.**
**Veh Dyn Test Data Managing** provides test report to **Requesting Department.**
**User** handles **Veh Dyn Data Managing** and **Veh Dyn Testing.**
**User** is operating the **Vehicle.**
**User** exhibits **Job Responsibility.**
**Job Responsibility** can be **Technologist, Test Engineer** or **Development Engineer.**

Figure 14. OPL of the Vehicle Dynamics Data Management System OPD of Figure 13

## Kinematics and Compliance

Actual analysis is performed on Kinematics and Compliance (K&C) data, obtained from the K&C laboratory. The K&C machine performs many tests on the Vehicle Dynamics Test Data. There are different types of K&C Data depending on the test type. These data will have to be analyzed by a test engineer using standard K&C analysis tools to generate metrics and plots. At the end of the K&C test battery, another program will create a summary report. Each specific test will generate metrics, such as steer, camber, etc. See Figure 15 is an OPD representation of the Kinematics and Compliance process, while Figure 16 is the corresponding OPL.

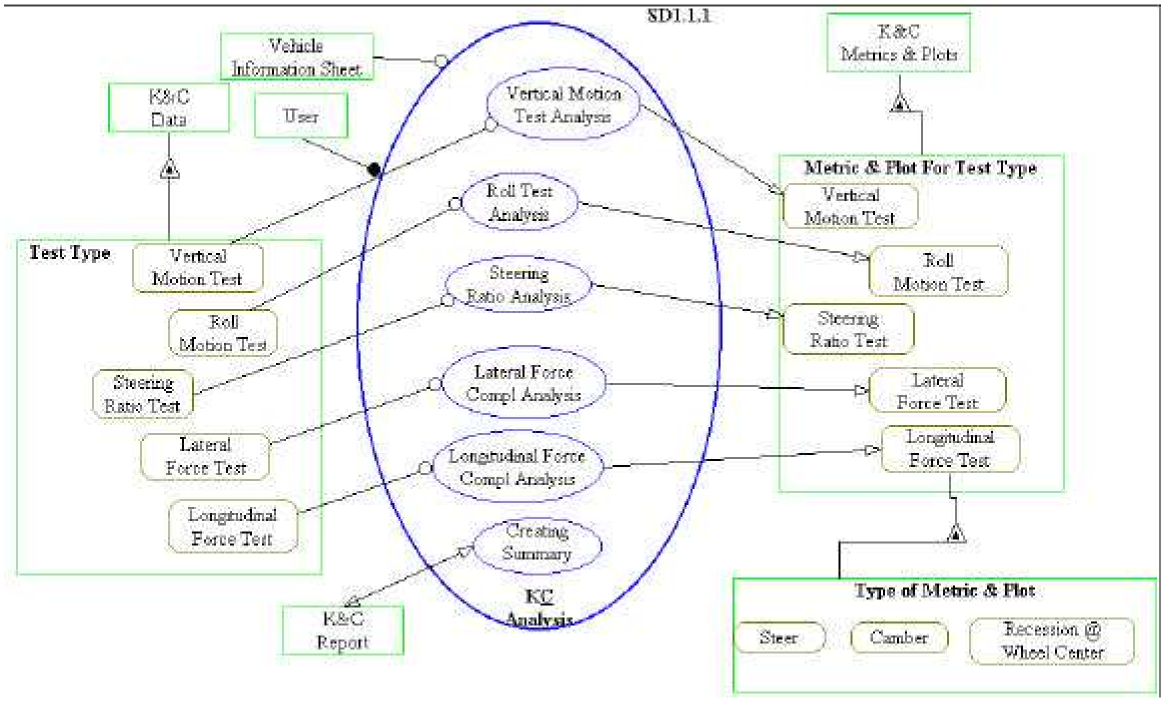

Figure 15. Kinematics and Compliance OPD

**K&C Analysis** expands into **SD1.1.1**
**K&C Data** exhibits **Test Type**
**Test Type** can be **Vertical Motion Test, Roll Motion Test, Steering Ratio Test, Lateral Force Test** or **Longitudinal Force Test.**
**K&C Analysis** requires **Vehicle Information Sheet.**
**User** handles **K&C Analysis.**
**Vertical Motion Test Analysis** requires **Vertical Motion Test Type K&C Data.**
**Roll Motion Test Analysis** requires **Roll Motion Test Type K&C Data.**
**Steering Ratio Analysis** requires **Steering Ratio Type K&C Data.**
**Lateral Force Compl Analysis** requires **Lateral Force Test Type K&C Data.**
**Longitudinal Force Compl Analysis** requires **Longitudinal Force Type K&C Data.**
**Creating Summary** affects **K&C Report.**
**K&C Metric & Plot** exhibits **Metric & Plot For Test Type**.
**Metric & Plot For Test Type** can be **Vertical Motion Test, Roll Motion Test, Steering Ratio Test, Lateral Force Test** or **Longitudinal Force Test.**
**Longitudinal Force Test** exhibits **Type of Metric & Plot.**
**Type of Metric & Plot** can be **Steer, Camber** or **Recession @ Wheel Center**

Figure 16. Kinematics and Compliance OPL

## Summary for Vehicle Dynamics Data Management

The Vehicle Dynamics Data Management application can be designed and developed in a traditional way by writing the operations and requirements in English prose. The design may be done using UML,[5] DFD or E-R Diagram[6] or some of the other OO methods surveyed by Henderson-Sellers[7] to represent the operations. The drawback is that most of the users and customers are not familiar with that language or representation. The following points are the highlights of the benefits OPM can bring to this particular project:

- OPM is an excellent way to represent daily activities, products, processes and other complex things.

- OPM has allowed representing the complete vehicle dynamics data management operation in a single model that specifies the systems function, structure and behavior aspects without sacrificing any clarity.
- OPM can be used as a common language to exchange design among members of a team.
- Since OPM design is visual and textual at the same time, it is easy to explain the design.
- OPL is very easy to generate from OPD.
- OPM will be a good tool for documenting the existing processes and as ISO documentation.

## Analytical Reliability and Robustness Computation Services

The methods and software associated with the Analytical Reliability and Robustness (AR&R) system are captured in the AR&R web site, available to Ford employees through the company's intranet. This site is a tool to support Ford's effort to improve and increase the amount of robust design and verification work done using computer models. The existing process and web site were generated in a somewhat ad hoc fashion as the original software tools were developed, thus no complete system specification exists and improvements continue to be made ad hoc.

### Current AR&R System

Ford engineers use the AR&R system to assess reliability and identify robustness improvements for components and systems that are designed using computerized models. These include CAE models, such as Finite Element Analysis, Adams modeling, Excel spreadsheet models, and models expressed explicitly as mathematical equations. The AR&R web site contains the following key sections that support application of the AR&R process:

- Home page: introductory material, organizational contacts, server selection choices, links
- Tutorial: detailed explanation of methods employed by the AR&R tools, including the equivalent of roughly 100 pages of text and examples organized into a web-based book with hyper-linked sections and chapters.
- Software page: source for five different web-based software packages that accept input from users and return results in text or html files. These packages include
  - Sampling Software (2 types)
  - Variability Assessment Software (2 types)
  - Response Surface Generation Software
- Reports Repository: knowledge base in which reports of actual studies using AR&R methods are stored, providing documentation and real-world examples to aid potential users.

### Requirements for Improved System

Current system users have made requests for several different types of improvement. These include

1. More complete documentation of the AR&R approach and tools
2. Faster result processing (through connections to faster servers or parallel processing)
3. More automation of the typical project through automatic linking of software outputs and inputs

Using OPM to model the system will help answer request 1 directly. OPM provides a concrete system specification that can be easily updated. OPM does not explicitly provide help in answering request 2, because this requires hardware upgrades. However, an OPM model of the interfaces required for a new server or a parallel processing scheme could prove very useful. Finally, OPM is an excellent tool for answering request 3 because it provides a clear map of the process and links that need to be implemented to create a more automated system.

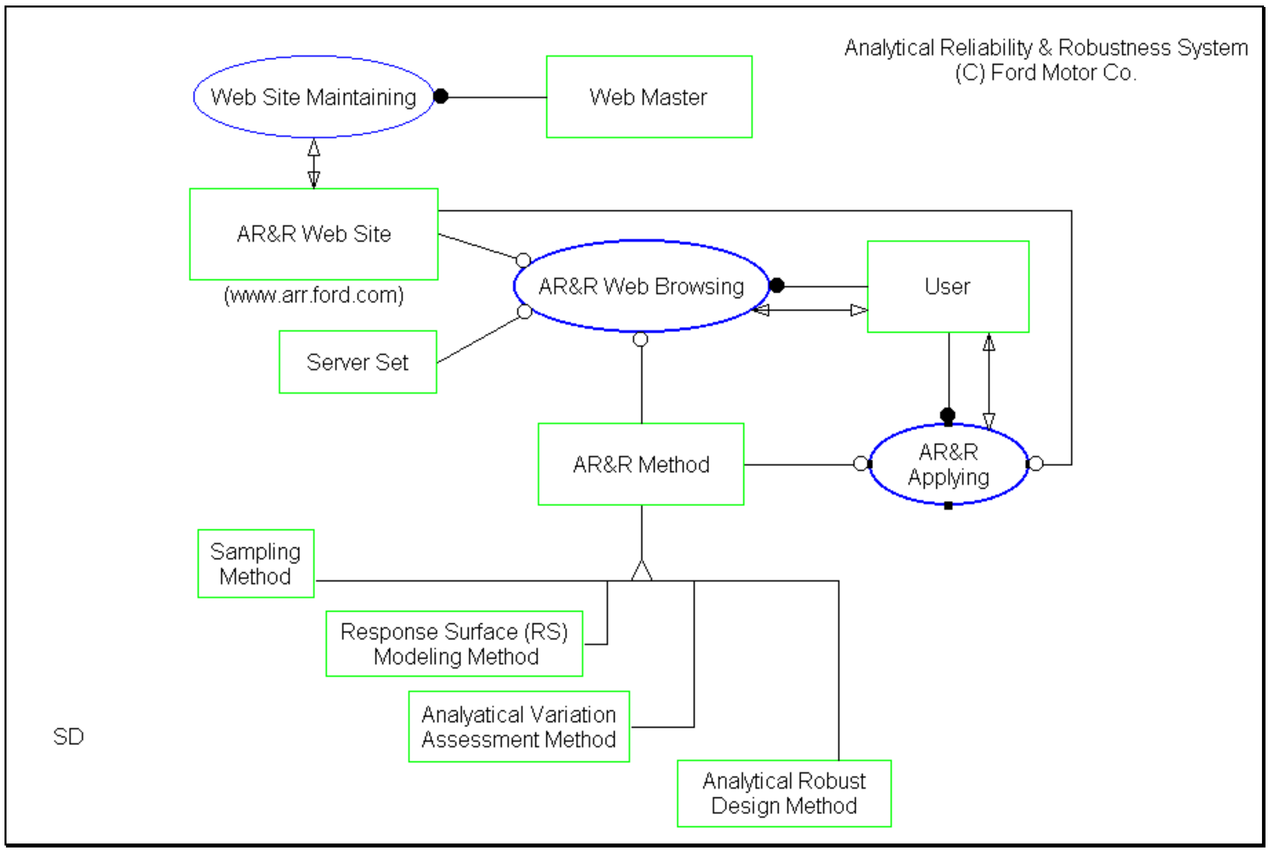

Figure 17. AR&R System Diagram

**Web Master** handles **Web Site Maintaining.**
**Web Site Maintaining** affects **AR&R Web Site**.
**AR&R Web Browsing** requires **AR&R Web Site, Server Set,** and **AR&R Method.**
**User** handles **AR&R Web Browsing** and **AR&R Applying.**
**AR&R Web Browsing** affects **User.**
**AR&R Applying** requires **AR&R Method** and **AR&R Web Site**.
**Sampling Method, Response Surface Modeling Method, Analytical Variation Assessment Method, and Analytical Robust Design Method** are **AR&R Method**s.

Figure 18. AR&R Top OPL paragraph

The top-level system diagram in Figure 17 and its OPL in **Error! Reference source not found.** show examples of the OPM modeling effort. These are followed by the AR&R applying process OPL and OPL in Figure 19 and Figure 20, respectively. The Problem Formulating process is elaborated in Figure 21and in Figure 22.

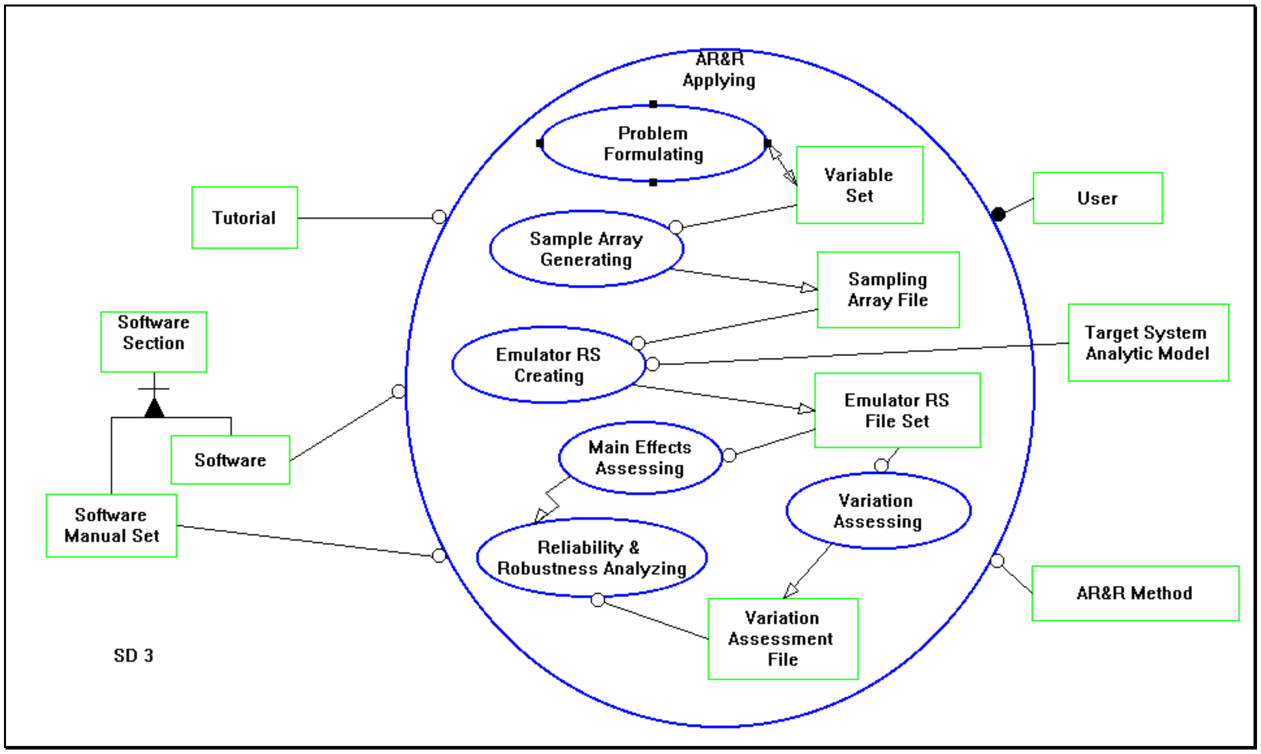

Figure 19. Zooming into the AR&R applying process

**Software Section** consists of **Software**, **Software Manual Set**, and more**.**
**AR&R Applying** requires **Tutorial, Software, Software Manual Set,** and **AR&R Method.**
**User** handles **AR&R Applying**.
**Problem Formulating** affects **Variable Set**.
**Sample Array Generating** requires **Variable Set.**
**Sample Array Generating** yields **Sample Array File.**
**Emulator RS Creating** requires **Sample Array File** and **Target System Analytic Model.**
**Emulator RS Creating** yields **Emulator RS File Set.**
**Variation Assessing** requires **Emulator RS File Set.**
**Variation Assessing** yields **Variation Assessment File.**
**Main Effects Assessing** requires **Emulator RS File Set.**
**Main Effects Assessing** invokes **Reliability & Robustness Analyzing.**
**Reliability & Robustness Analyzing** requires **Variation Assessment File.**

Figure 20. OPL of the AR&R applying process

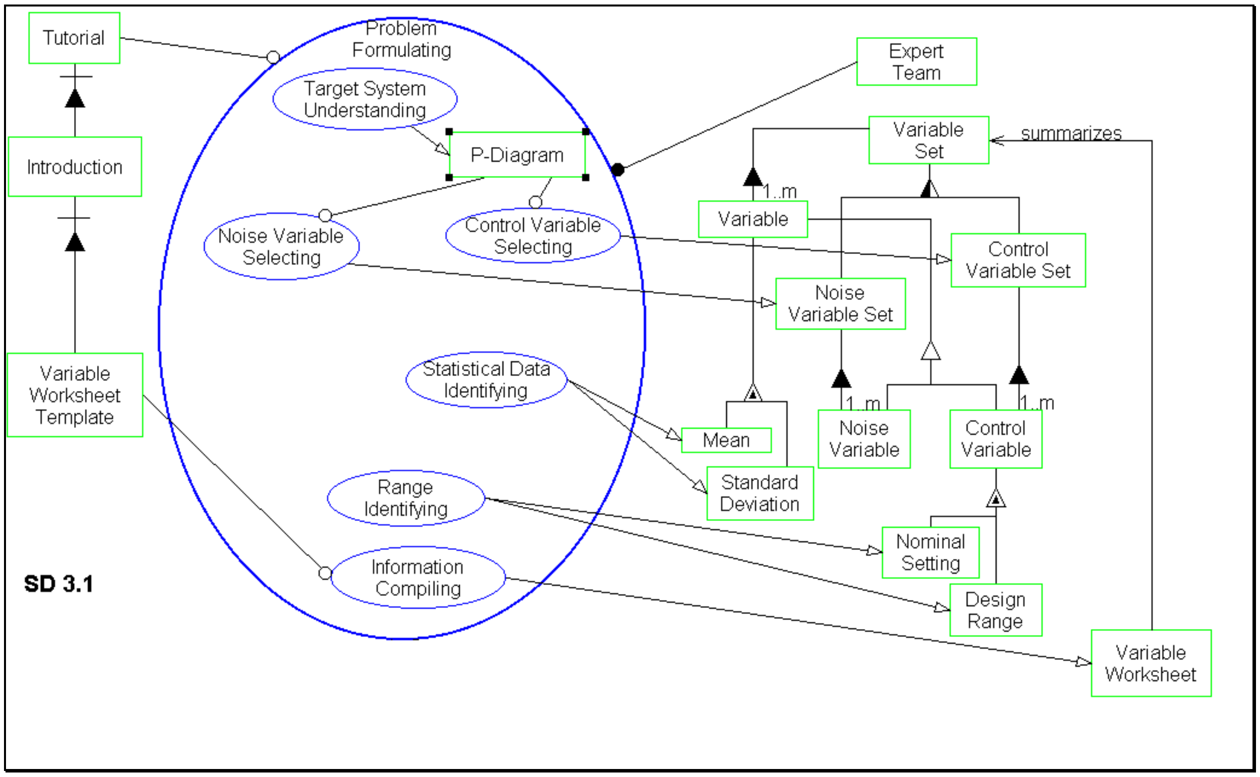

Figure 21. The Problem Formulating process

**Tutorial** consists of **Introduction** and more.
**Introduction** consists of **Variable Worksheet Template** and more.
**Problem Formulating** requires **Tutorial.**
**Expert Team** handles **Problem Formulating.**
**Target System Understanding** yields **P-Diagram.**
**Noise Variable Selecting** requires **P-Diagram.**
**Control Variable Selecting** requires **P-Diagram.**
**Control Variable Selecting** yields **Control Variable Set.**
**Noise Variable Selecting** yields **Noise Variable Set.**
**Statistical Data Identifying** yields **Mean** and **Standard Deviation.**
**Range Identifying** yields **Nominal Setting** and **Design Range.**
**Information Compiling** requires **Variable Worksheet Template.**
**Information Compiling** yields **Variable Worksheet.**
**Variable Worksheet summarizes Variable Set.**
**Variable Set** consists of one or more **Variables.**
**Variable Set** consists of and generalizes **Noise Variable Set** and **Control Variable Set.**
**Control Variable Set** consists of one or more **Control Variables.**
**Noise Variable Set** consists of one or more **Noise Variables.**
**Control Variable** exhibits **Nominal Setting** and **Design Range.**
**Noise Variable** and **Control Variable** are **Variables.**
**Variable** exhibits **Mean** and **Standard Deviation.**

Figure 22. OPL of the Problem Formulating process

## Summary for AR&R System Modeling using OPM

The main benefit of OPM is its ability to identify system objects, processes, and the relationships among them in a structured way. The resulting OPD set becomes an excellent framework for identifying how to implement structural and procedural improvements. The resulting OPL script provides a well-defined set of existing and future specifications for the system. The aility to freely switch from text to graphics and back is of great value to understanding the system as a whole with a single graphic and textual model, without the need to consult various models and carry out mental transformation among these various models.

## Summary and Conclusions

A fundamental issue in making computing pervasive concerns overcoming the barrier to application generation that current programming languages pose for non-programmer domain experts. Combining natural language and intuitive graphics, Object-Process Methodology (OPM) is a system development and specification approach that integrates the major system aspects - function, structure and behavior - within a single diagrammatic and textual model. Having applied OPM in a variety of domains, this paper introduces OPM as a vehicle to enhance pervasive computing in the design of IT, automotive and aerospace systems by presenting the analysis and design of five case studies. The common thread among these cases is the fact that all demonstrate that OPM provides a natural way for people from various disciplines and backgrounds to communicate system-related ideas, designs and analyses. What these case studies represent is how customers, business people, and engineers can pick up the notation, speak the same language and communicate freely to develop and understand the system. Moreover, OPM provides a standard natural language for specifications as a complementary modality beside the OPD diagramming technique. This unique feature does not exist in other techniques such as UML. Indeed, an empirical study has shown[8] that a single model method is significantly superior over a multi-model one.

The specification results in a set of Object-Process Diagrams and a corresponding equivalent set of Object-Process Language sentences. The synergy of combining structure and behavior within a single formal model, expressed both graphically textually yields a highly expressive system modeling and specification tool. The comprehensive, unambiguous treatment of the system at the semantic level is formal, yet intuitive and clear, suggesting that OPM is a prime candidate for becoming a common standard platform for a generic-domain-independent systems development. This, in turn, will foster computing pervasiveness, as non-programmer domain experts will be able to specify systems naturally and intuitively without the need for mediating application programmers.

### Benefits and Drawbacks

OPM provides a framework for understanding precisely how the system works. This understanding is revealed gradually and iteratively as OPDs are developed, refined, and expanded into new OPDs in an ongoing cycle. OPL is an orthogonal modality that enhances expressive power and fosters clear understanding of the system at all levels. At some point, continuing the cycle yields diminishing returns, and it is time to stop modeling the existing system.

Current drawbacks to OPM stem from the fact that it is relatively unknown. Any modeling of a system must be accompanied by a tutorial in OPM in order to communicate the model. Another drawback is the lack of connections to existing tools and methods already used to model and develop systems. OPM has the potential to be a common "translating mechanism" between these tools and methods, but that potential is yet to be realized. Hopefully, both these drawbacks can be overcome with time as OPM becomes common in systems design and interest increases in application development.

### Directions for Future Work

OPM is a coherent, consistent development paradigm, yet is still in its infancy with respect to wide adoption for application. Making OPM more immediately useful to systems designers and developers is a key need. Some questions to direct future work in this area include

- Are there heuristics or guidelines for where to begin and what path to follow in modeling a system?
- How does one know when to stop the iteration of OPD development, e.g., when does the work of diagramming well-known structures in system reach the point of diminishing returns?
- Can a library of basic structures be developed for use as templates by system designers?
- Are there ways to use OPDs and OPL to measure whether a particular system design is good, or to identify potential weaknesses in a design?
- Can complexity be quantifies with OPM?
- Could a more structured process be developed for turning existing requirement documents into OPL script?
- How can OPM be extended naturally to model behavior of systems over time or provide dynamic system simulation? Research in this direction is in progress.[9]
- How can OPDs and OPL script be translated into programs and diagrams immediately useful to system builders and recognized by others in their community? (E.g., design structure and relationship matrices, axiomatic design matrices, requirement cascades, reliability block diagrams, systems dynamics, fault trees, etc.)

## References

[1] Dov Dori, Object-Process Analysis: Maintaining the Balance between System Structure and Behavior. Journal of Logic and Computation, 5, 2, pp. 227-249, 1995.

[2] Dov Dori, Object-Process Methodology: A Holistic Systems Paradigm, Springer Verlag, Heidelberg, New York, 2000 (in press).

[3] http://command.mit.edu/16.982/index.html

[4] http://sdm.mit.edu/home1.nsf/home

[5] Fowler, M. UML Distilled, Second Edition, Addison Wesley, Reading, MA, 1999.

[6] Chen, P.P., 1976. The Entity Relationship Model – Toward a Unifying View of Data. ACM Trans. on Data Base Systems 1, 1, pp. 9-36.

[7] Henderson-Sellers, B. and Bulthuis, A. Object-Oriented Metamethods, Springer,-Verlag, New York, 1997.

[8] M. Peleg and Dov Dori, The Model Multiplicity Problem: Experimenting with Real-Time Specification Methods. IEEE Transaction on Software Engineering, 26, 12, pp. 1-18, 2000 (in press).

[9] M. Peleg and D. Dori, Extending the Object-Process Methodology to Handle Real-Time Systems. Journal of Object-Oriented Programming 11, 8, pp. 53-58, 1999.