

Conference on Systems Engineering Research (CSER'13)

Eds.: C.J.J. Paredis, C. Bishop, D. Bodner, Georgia Institute of Technology, Atlanta, GA, March 19-22, 2013.

# Extending Design Capabilities of SysML with Trade-off Analysis: Electrical Microgrid Case Study

Dimitrios Spyropoulos, John S. Baras\*

*Institute for Systems Research, and  
Department of Electrical and Computer Engineering  
University of Maryland, College Park, MD 20742, USA*

---

## Abstract

Changes in technology, economy and society create challenges that force us to rethink the way we develop systems. Model-Based Systems Engineering is an approach that can prove catalytic in this new era of systems development. In this paper we introduce the idea of the “modeling hub” in order to realize the vision of Model-Based Systems Engineering and especially we focus on the trade-off path of this hub. For that purpose the design capabilities of SysML are extended by integrating it with Consol-Optcad, a powerful multi-criteria optimization tool for trade-off analysis. The integration and its implementation is applied to analyze a multi-criteria optimization problem concerning power allocation and scheduling in a microgrid.

© 2013 The Authors. Published by Elsevier B.V.

Selection and/or peer-review under responsibility of Georgia Institute of Technology

*Keywords:* modeling hub; trade-off analysis; metamodels; model transformations; system design;

---

## 1. Introduction

Last decade we entered a new era where systems complexity has increased dramatically. Complexity is increased both by the number of components that are included in each system as well as by the dependencies between those components. Moreover, today, systems tend to be more software dependent and that is another challenge that engineers and people involved in the development of such systems, face. The challenge is even greater when a safety critical system is considered, like an airplane or a passenger car. There is a need for development of software that is provably error-free. Moreover, when software dependent systems interact also with the physical environment then we have the class of cyber-physical systems (CPS). The challenge in CPS is to incorporate the inputs from the physical environment in the logic of the embedded software. Nowadays, more frequently we observe systems that cooperate to achieve a common goal, even though there were not built for that reason. These are called systems of systems. For example, the Global Positioning System (GPS) is a system by itself. However, it needs to cooperate with other systems when the air traffic control system of systems is under consideration. The analysis and

---

\* Corresponding author. Tel.: +1-301-405-6606; fax: +1-301-314-9218.

*E-mail address:* [baras@umd.edu](mailto:baras@umd.edu).

development of such systems should be done carefully because of the emergent behavior that systems exhibit when they are coupled with other systems. However, apart from the increasing complexity and the other technical challenges, there is a need to decrease time-to-market for new systems as well as the associated costs. This specific trend and associated requirements, which are an outcome of global competitiveness, are expected to continue and become even more stringent.

As it can be easily understood, due to these challenges and market pressure the whole process of how systems are developed is going to change inevitably in a dramatic way. There is a need for rigorous and quantitative ways to understand, analyze and develop systems. Percentages of systems that fail or have cost and schedule overruns confirm this need. Model-Based Systems Engineering (MBSE) has emerged as a promising approach to this crucial for the economy and the society problem, and has found a lot of applications lately in industry. SysML [1] was developed in order to facilitate MBSE. The appeal of SysML is that it provides systems engineers with a high-level visual representation of design concerns, like system requirements, system structure and behavior, and system parameter and performance metrics. Something that SysML does not provide is a way for engineers to formally evaluate and rank design criteria, conduct sensitivity analysis, search design spaces for optimal design solutions, and conduct trade studies. To address this challenge we have worked on the concept that SysML needs to be integrated with industrial-strength multi-objective algorithms, constraint-based reasoning algorithms, with appropriate linkages to modeling/simulation environments. In this paper we present the integration between SysML and Consol-Optcad, which is a sophisticated multi-criteria optimization tool developed at the University of Maryland. This integration focuses on the trade-off part of the system modeling hub, which we use as a roadmap for our overall MBSE process. □ □

The rest of this paper is organized as follows. Section 2 is devoted to related work. In section 3 we provide information on Model-Based Systems Engineering and on the SysML language. In section 4 we describe the modeling hub and analyze in detail the trade-off part, while section 5 analyzes the integration implementation. In section 6 we use the integration to perform trade-off analysis on an electrical microgrid. Finally, section 7 concludes the paper and provides insights on future work.

## 2. Related Work

In recent literature, there are approaches on the integration of SysML with external analysis tools and mathematical solvers. The most popular is Paramagic that offers integration between SysML and Modelica, Matlab and Mathematica [2]. However, these approaches lack support for performing trade-offs and design space exploration. Consol-Optcad is a strong multi-criteria optimization tool with innovative visualization techniques that help engineers understand the impact of design choices and perform design space exploration. Recently InterCAX LLC and Georgia Institute of Technology added trade-study capabilities in Paramagic [3], but still the capabilities regarding design space exploration and reaching optimal solutions are limited. A more interesting approach is the one proposed by Min et al [4]. They integrated SysML with Model Center, which provides existing integrations to a large number of analysis tools, allowing users to handle multiple heterogeneous analyses in the same environment. Model Center supports multiple optimization algorithms and allows for design space exploration. On the other hand, it lacks unique features of Consol-Optcad such as interaction with the user while the optimization is in progress, and the ability to handle functional objectives/constraints (specifications depending on a continuous range of a parameter(s)).

There are also other powerful tools that are not integrated with SysML, like the IBM CPLEX, commercially available from IBM-ILOG. While not incorporating all the features of Consol-Optcad, it offers a lot more capabilities and it is widely used in industry. Therefore, our long-term goal is to integrate CPLEX and the IBM-ILOG constraint solver, as the main trade-off tool in our system modeling hub.

In this paper we also provide a broader perspective to the integration by incorporating within it the concept of the system modeling hub, which provides a clear framework for the development of a holistic MBSE environment. The SLIM environment proposed by Bajaj et al [3] is based on a similar perspective, however the present paper provides a clear methodology for the integrations, which is independent of specific platforms and tools.

Finally in this work, the Enterprise Architect eMoflon plugin for model transformations was used for the first time for an integration of SysML with an external tool.

### 3. Background

#### 3.1. Model-Based Systems Engineering

“Model-Based Systems Engineering (MBSE) is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases” [5]. In contrast with the traditional document-centric Systems Engineering approach, here the models are the main product of each process and are used for the communication between the different teams that take part in the development. In Fig. 1 [6] the core steps of the MBSE process are illustrated.

For each system the starting point is the available information. Afterwards, the initial system requirements are defined together with the desired measures of effectiveness (MoE). The MoE are used at the trade-off stage as criteria for the selection of the best system configuration. After the requirements phase, the models of behavior and structure are developed. Then the process continues by mapping the specified system behavior to the structure. During the MBSE process derivative requirements are generated and thus, if needed, changes to system configuration are performed. After exploring the design space and selecting the best alternative the system shall be validated and verified. This phase is crucial, as it makes sure that all requirements are satisfied and that the system meets its goals.

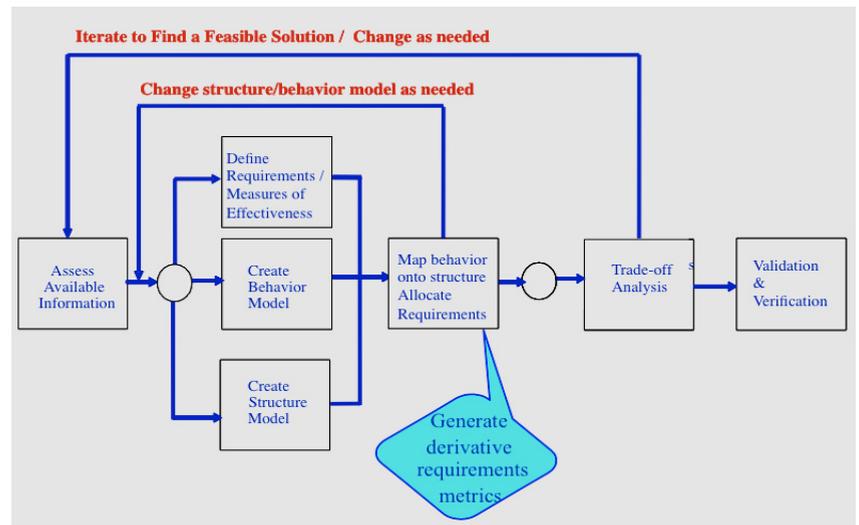


Fig. 1. The MBSE process [6]

While keeping all the advantages of Systems Engineering that help in reducing complexity and better manage a system development project, MBSE offers even more capabilities. The greatest advantage of MBSE is that separates design from production. That is a major change to the existing status in systems development. Each company could have an “artificial” product made by using only models and then let other companies compete for the actual production. That will consequently drive costs down and possible reduce time to market. The same method was used in VLSI design and production with phenomenal success. The use of models allows for faster and more rigorous communication between engaged teams and stakeholders. Time overhead to manage system documentation will be extinct. Fewer errors will come up due to misunderstandings or oversights, mostly due to the more formal semantics that models offer. Another characteristic of MBSE is re-usability of models. A subsystem or a component model can be used in many systems avoiding the need to be developed from scratch every time it is included in a new system.

#### 3.2. SysML

SysML [1] is a general purpose graphical modeling language that was developed based on UML and is a key enabler for the MBSE process by providing ways for the representation and analysis of complex engineering systems. SysML supports the specification, analysis, design, verification, and validation of systems that include hardware, software, data, personnel, procedures, and facilities. The four main parts of a SysML model are the structure, behavior, requirements and parametric diagrams. SysML introduced the requirements diagram to better represent and handle requirements. Apart from representing the requirements, the designer can specify their decomposition and also can allocate low-level requirements to parts of structure and behavior. Consequently, the requirements diagram offers better traceability among requirements and system components. Parametric diagrams are used to specify equations that characterize the system and link them to system component properties. Moreover,

parametric diagrams are the key to linking SysML to analysis models, including trade-off analysis models such as those provided by multi-criteria optimization and constraint-based reasoning tools.

#### 4. The Modeling Hub

As described in section 3 a major challenge in MBSE is to have models that are consistent with each other. However, besides having consistent data there is a need for the models to work together in order to offer a holistic Systems Engineering approach to the designer. In this work the concept and construct of “modeling hub” is introduced to deal with this challenge.

##### 4.1. Hub Architecture

As it was mentioned in the introduction, SysML, as a language for describing the high-level system architecture, can act as a catalyst for the integration of various modeling environments, as well as analysis/design environments, for complex systems. Therefore, SysML is used in the core of the modeling hub (Fig. 2). The main aim is to integrate this core module with external tools, each one used in a different phase of the Systems Engineering development process [7]. The resulting MBSE environment can be thought of as a “virtual” product line management (PLM) environment, across discipline tools. To achieve this integration a three-layer approach needs to be followed. Initially, for the tool we need to integrate, a domain specific profile is created in SysML. Then a model transformation is defined, followed by the implementation of tool adapters that are used as a middleware for exchanging information between the model transformation layer and the other components of the hub. Fig. 2 presents these layers as well as the areas for which we need to integrate tools with the core module to realize the MBSE vision of a holistic system design experience.

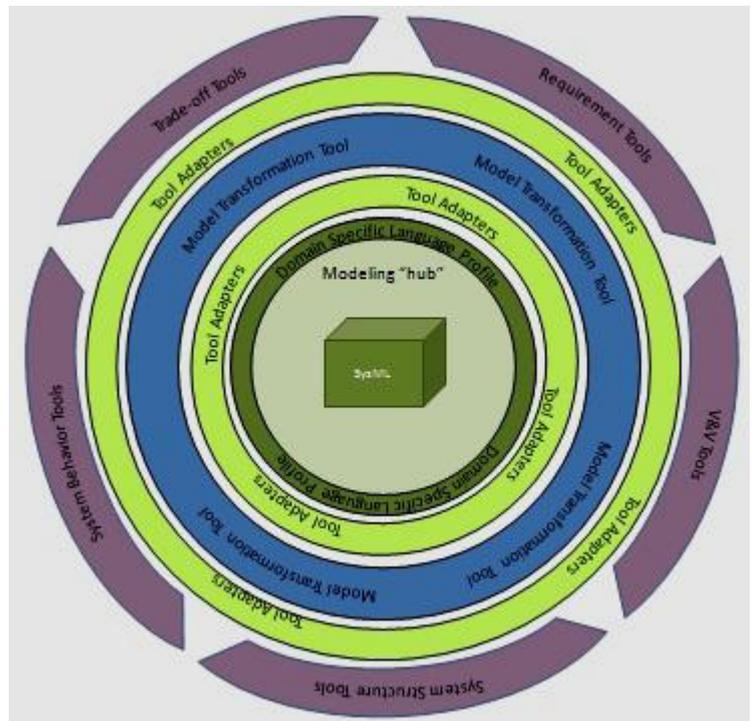


Fig. 2. The modelling hub

Fig. 2 presents these layers as well as the areas for which we need to integrate tools with the core module to realize the MBSE vision of a holistic system design experience.

##### 4.2. Trade-off Analysis Path

Trade-off is an essential part of system design, as it implements design space exploration. An integration of SysML with a trade-off tool will allow the designer to make decisions faster and with more confidence. The probability of human error during transfer of data between tools will not exist and consequently the overall quality and performance of the system will be increased. This integration becomes even more crucial when we think of today's systems that have multiple competing objectives to satisfy and a lot of design parameters.

In this work the focus is on integrating SysML with Consol-Optcad, which is a trade-off analysis tool that was developed at the University of Maryland. Consol-Optcad is a multi-objective optimization tool that allows interaction between the model and the user. It can handle non-linear objective functions and constraints with continuous values. Another version of Consol-Optcad has been developed to handle also logical variables, via integer and constraint programming [8]. In systems development and after the system structure is defined there is a need to calculate the design parameters that best meet the objectives and constraints. Usually when we deal with complex systems and optimization is under consideration, this is not a trivial task. The support of an interactive tool,

like Consol-Optcad, to help the designer resolve the emerging trade-offs is necessary. A major advantage of Consol-Optcad is that it allows the user to interact with the tool, while the optimization is under way. The designer might not know or might not be in a position at the beginning to specify what optimal design means. Therefore such interaction with the tool could be of great benefit [9, 10]. Another key feature of Consol-Optcad is the use of the Feasible Sequential Quadratic Programming (FSQP) algorithm for the solver. FSQP's advantage is that as soon as we get an iteration solution that is inside the feasible region, feasibility is guaranteed for the following iterations as well. Moreover, very interesting is the fact that besides traditional objectives and constraints Consol-Optcad allows the definition of functional constraints and objectives that depend on a free parameter. Consol-Optcad has been applied to the design of flight control systems [11], rotorcraft systems [12], integrated product process design (IPPD) systems [8] and other complex engineered systems.

## 5. SysML Integration with Consol-Optcad

This section describes in detail the integration framework and the separate steps that were followed to achieve the integration between SysML and Consol-Optcad. Actually here is a good point to mention that MagicDraw was the SysML tool that was used for this integration. SysML is not a tool specific language, but MagicDraw was used because it is more open than other tools and it can be modified more easily. To simplify things from now on when integration with SysML is mentioned, we refer to integration with MagicDraw SysML.

### 5.1. Integration Framework

Fig. 3 presents the architecture of the integration together with numbered steps that need to be followed to complete the integration process. According to the three-layer approach of section 4, the integration process is divided into three main parts. The first part concerns the mapping of the objects between the two languages (SysML, Consol-Optcad). It also includes the development of specific semantics that are used for that purpose; in this case a profile of Consol-Optcad in SysML is created. The second part is the meta-modeling layer where the transformation between the two models takes place. The last part consists of implementing the appropriate tool adapters.

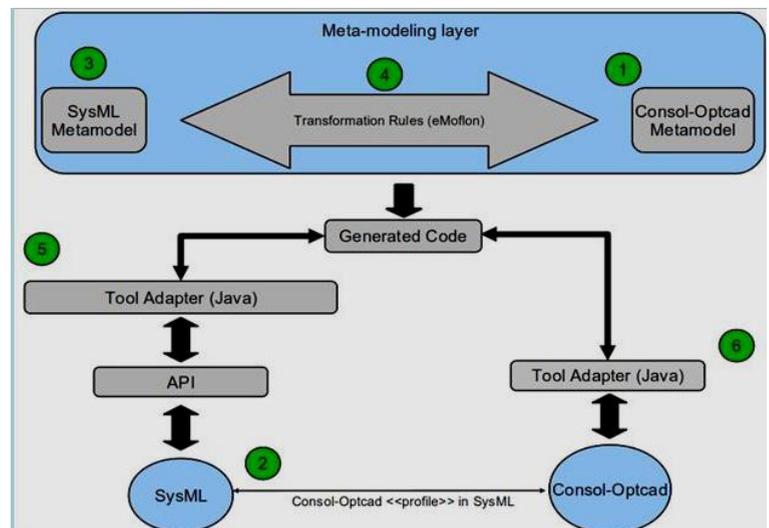


Fig. 3. Integration Framework

### 5.2. Consol-Optcad Profile in SysML

The creation of a Consol-Optcad profile is the first step of the integration process. Profiling is the mechanism that SysML has to allow the designer to use additional constructs inside the development environment. After a profile is being built and since it gives the user the ability to use constructs of a specific tool directly in SysML, it decreases significantly the design effort. A SysML Profile is composed by a set of stereotypes and their relationships [13]. Each Consol-Optcad construct is represented in the profile diagram by a stereotype, according to the Consol-Optcad specification document [8]. After the profile has been created the designer can load the new profile in the project and start using it by simply dragging and dropping Consol-Optcad constructs in the block definition diagram area.

### 5.3. Meta-modeling Layer

The meta-modeling layer is the second major part of the integration process. A meta-modeling layer stands one abstraction layer above the actual design implementation in a modeling language. A meta-model consists of the

constructs of a modeling language together with the rules that specify the allowable relationships between these constructs. It can be considered as the grammar of that modeling language. At the meta-modeling layer model transformations take place. In general a model transformation is the conversion of a model from one language to the other. There are many alternatives in terms of model transformation tools, like ATL, GME, eMoflon, QVT. In this research the eMoflon model transformation tool was used which is developed at TU Darmstadt [14, 15]. The eMoflon tool was chosen instead of other meta-modeling tools due to a number of reasons. First of all, graph transformations is the underlying theory for model transformations, a fact that makes the semantics strong and can lead to satisfaction of formal properties such as correctness, completeness and termination. The eMoflon toolsuit allows for graphical representation of meta-models and transformation rules, a fact that makes the model transformation process less cumbersome and less error prone. The eMoflon can also generate automatically Java code for the model transformations. Since most of the tools today are implemented in Java, eMoflon code for transformations can be easily used and integrated in other tools. Finally, the development environment is user friendly, it is well documented and it has a strong support/development team.

**Meta-models:** The first step towards making a model transformation is to define the meta-models of the languages of interest. Both meta-models were developed by using the eMoflon Enterprise Architect (EA) plug-in and they follow the Ecore format. The Consol-Optcad meta-model was created from scratch according to its specification [8], while the Ecore meta-model of SysML is available online.

**Transformation Rules:** The second step is to define the transformation rules, expressed in eMoflon via Story Diagrams (SDMs), which constitute a graph grammar language and provide a mechanism for defining unidirectional graph transformations. SDMs adopt concepts from UML class, activity and collaboration diagrams [16].

As described in [14] a rule  $r:(L,R)$ , defined by an SDM follows the process below when is applied to a graph:

- Find a match  $m$  for the precondition  $L$  in  $G$
- Delete all the elements that are present in the precondition but not in the post-condition ( $\text{Destroy} := (L \setminus R)$ ), to form  $(G \setminus \text{Destroy})$
- Create new elements that are present in the post-condition but not in the precondition ( $\text{Create} := (R \setminus L)$ ), to form a new graph  $H = (G \setminus \text{Destroy}) \cup \text{Create}$

The nodes of the graph can be considered as blocks of a system while the edges are the relationships that connect them. After defining all the SDMs the EA models are exported to an Eclipse project.

#### 5.4. Tool Adapters

Tool adapters work as the “glue” between the different pieces of software. Their role is to access/change information inside a model and also call the appropriate Java functions generated by the eMoflon tool to perform the model transformations. All adapters for this project were developed inside a single MagicDraw plug-in, which is the way to extend the functionality of MagicDraw. The plug-in is the core of the software part of the integration process. It was developed using Java; for the development the Eclipse platform was used [17]. When the plug-in is invoked the SysML model is transformed automatically to a Consol-Optcad Problem Description File, a text file that contains the problem formulation description for the Consol-Optcad environment.

## 6. Trade-off Analysis of an Electrical Microgrid

In section 5 we described the integration methodology and its implementation. In this section the aim is to illustrate the way the integration works through an example: trade-off analysis in an electrical microgrid.

### 6.1 A Microgrid and its components

The typical way of producing and distributing power is through a centralized power system. This system has served well humanity during the last century, but it has some important inefficiencies [18]. To address these shortcomings the notion of Distributed Generation (DG) has emerged. In DG the generating systems are of small scale, their use is local and they are geographically distributed. However, DG can cause problems to the network, like reverse power flow, excessive voltage rise, increased fault levels, harmonic distortion and stability problems, due to their independent operation. To overcome such problems various distributed energy resources (DERs) are

grouped together and together with loads to form what is called a microgrid. The Energy Management System plays a central role in the smooth operation of microgrids; it makes the decisions about generation and distribution of electrical energy. These decisions are based on many factors, like power demand, weather, price of electricity and heat, fuel cost, emissions cost and government policies, to name a few. The DERs that take part in a microgrid can be electrical, thermal or a combination. Solar panels, small wind and hydro generators, micro turbines, diesel engines, fuel cells, gas turbines are some examples of DERs.

6.2 Problem Formulation

Table 1. Characteristics of the power sources

Power Source	Capacity (kW)	Efficiency %	NO <sub>x</sub> (lb/MWh)	CO <sub>2</sub> (lb/MWh)	SO <sub>2</sub> (lb/MWh)	K <sub>OM</sub> (\$/kWh)
Phosphoric Acid Fuel Cell	25	37	0.03	1,078	0.0006	0.00419
SCR Controlled Diesel Engine	1000	38	4.7	1,432	0.454	0.01258
Micro Turbine	25	25	0.44	1,596	0.008	0.00587

We define a microgrid that consists of three power sources: one microturbine, one fuel cell and one diesel engine. The characteristics of each type of power source are listed in Table 1, with data from [19, 20, 21]. The microgrid is supposed to provide power to a residential building that has 50 apartments. We would like to find an optimal solution in terms of scheduling and power output of each engine for a period of 24 hours. The optimal solution is sought while trying to minimize operational cost, fuel cost, emissions and meet customer demand. We assume that each power source can be turned on and off only two times during a day, because of the costs associated with turning on/off power sources. We define 5 design variables for each power source:

- $P_i$  ( $i = 1, 2, 3$ ) represents the output power of each power source
- $t_{i\_on1}$  ( $i = 1, 2, \dots, N$ ) represents the first time that power source  $i$  starts to operate
- $t_{i\_off1}$  ( $i = 1, 2, \dots, N$ ) represents the first time that power source  $i$  is turned off
- $t_{i\_on2}$  ( $i = 1, 2, \dots, N$ ) represents the second time that power source  $i$  is turned on
- $t_{i\_off2}$  ( $i = 1, 2, \dots, N$ ) represents the second time that power source  $i$  is turned off

The objectives and the demand constraint are described below:

**Operational Cost:** This objective aims to minimize the total operation and maintenance microgrid cost [22]:

$$OM(\$) = \sum_{i=1}^N K_{OMi} P_i t_{i\_operation}$$

where  $N$  is the number of generating power sources,  $K_{OMi}$  is a constant for each power source defined in Table 1,  $t_{i\_operation}$  (h) is the total time period that power source  $i$  is ON and  $P_i$  (kW) is the power output of each source.

**Fuel Cost:** Another objective is to minimize fuel cost, calculated by the following formula [22]:

$$FC(\$) = \sum_{i=1}^N C_i R_i \frac{P_i t_{i\_operation}}{n_i}$$

where  $N$  is the number of generating power sources,  $C_i$  (\$) the price of fuel that each source utilizes,  $R_i$  (gallon/kWh) the consumption rate of each power source,  $P_i$  (kW) is the power output of each source,  $t_{i\_operation}$  (h) is the total time period that power source  $i$  is ON, and  $n_i$  the efficiency of each power source.

**Emissions:** Emissions should also be minimized and they are calculated using the following formula [19]:

$$EC(\$) = \sum_{i=1}^N \sum_{k=1}^M \alpha_k (EF_{ik} P_i t_{i\_operation} / 1000)$$

where  $N$  is the number of generating power sources,  $M$  the number of emission types,  $\alpha_k$  (\$/lb) a constant showing the cost of emission  $k$ ,  $EF_{ik}$  (lb/MWh) is the emission factor of power source  $i$  and emission type  $k$ ,  $t_{i\_operation}$  (h) is the total time period that power source  $i$  is ON and  $P_i$  (kW) is the power output of each source.

**Meet Demand:** Meet demand is a functional constraint that corresponds to the power demand in the fifty apartment residence of the example each time of day. Time is the free parameter and can take values from 1 to 24. Data from [23] were used to define an approximate function for the power demand (where  $t$  is time):

$$Demand(kW) = 50 \cdot (-0.6 \sin(\frac{\pi t}{12}) + 1.2)$$

There are also three constraints that ensure the correct operation of each generating unit:

- $t_{i\_off1} - t_{i\_on1} \geq x_i$  and  $t_{i\_off2} - t_{i\_on2} \geq x_i$ . When a generating power source is turned ON, it shall remain ON for at least  $x_i(h)$  time units. If that constraint is not met then the power sources may malfunction.
- $t_{i\_on2} - t_{i\_off1} \geq y_i$ . When a generating power source is turned OFF, it shall remain OFF for at least  $y_i(h)$  time units. If that constraint is not met then the power sources may malfunction.

The problem has a total of 15 design variables, 10 constraints and 3 objective functions.

### 6.3 Problem Solution using the SysML Consol-Optcad Integration

In this section a solution to the microgrid problem using our integration will be presented; including both the SysML model and the trade-off analysis part in Consol-Optcad, after the transformation is being performed.

**SysML Model of the Problem:** The first step towards a solution is to build in SysML the microgrid system structure and specify the trade-off analysis problem, utilizing the constructs offered by the Consol-Optcad profile (subsection 5.2). The SysML block definition diagram was used to specify the structure of the microgrid system,

which consists of three DERs together with their attributes. The attributes represent the output power of each source and the times the source goes on and off. The trade-off analysis problem model was specified in SysML by using block and parametric diagrams. Each design parameter of the trade-off model obtains its initial value from the instance diagram. This value is transferred through the parametric diagram. Fig. 6 shows how the process of passing parameters is modeled. Besides design variables, the objectives and constraints that take part in the trade-off study were modeled in SysML. When the whole modeling process in SysML is completed the designer can make the transformation and start using Consol-Optcad.

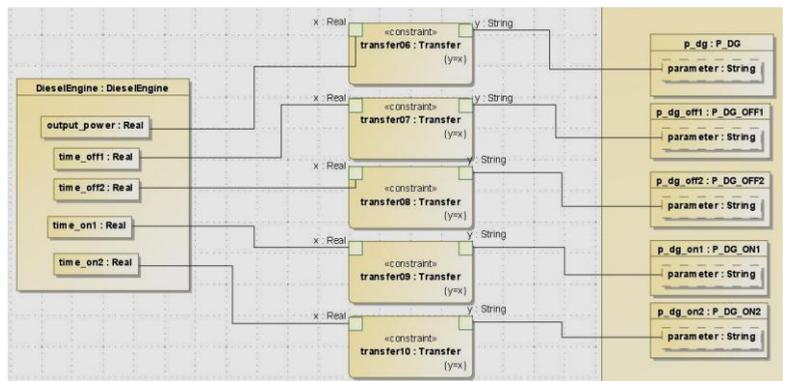


Fig. 6. Example of a Parametric Diagram for the microgrid system

**Solving the Problem in Consol-Optcad:** After modeling the system and the trade-off analysis in the MagicDraw SysML environment the integration mechanism can be used to automatically transform the existing model in a problem description file inside Consol-Optcad. For the multi-objective problem that is under consideration, we run the FSQP algorithm of Consol-Optcad several times with different initial conditions. Below, the best solution found for the examined initial conditions, is presented with a series of screenshots from the Consol-Optcad environment.

**Iteration 1 (Initial Phase):** Fig. 7 shows the performance comb (pcomb) after the first iteration. Pcomb is the structure that Consol-Optcad uses to present to the user the results of the optimization process at each iteration. Pcomb includes information on the current value of an objective or a constraint and shows if that value satisfies the specified limits. Those limits represent good and bad values that were set by the user and they are marked in pcomb by vertical lines. From the pcomb it can be seen that one hard constraint is not satisfied. The normalized value of that constraint is on a red circle and the constraint is not met because that value is on the right side of the vertical line that represents the good value. A hard constraint shall strictly have a value above the good value limit while a soft constraint shall be at least above the bad value limit. All other hard constraints and objectives are satisfied. However, the functional soft constraint that represents the need to meet the energy demand is not satisfied. This is shown on pcomb (red dot), but is clearer in Fig. 8 where the functional

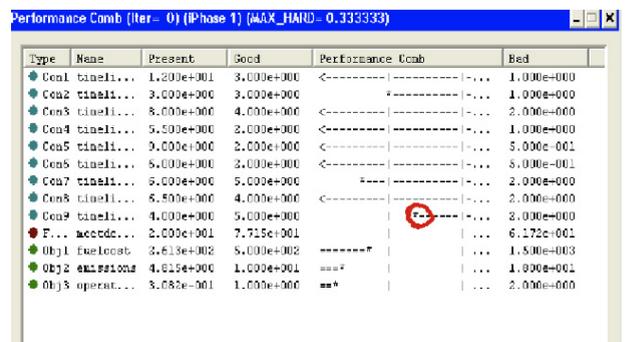


Fig. 7. Pcomb after the first iteration

constraint curve (blue) is below the good (green) and the bad (red) curves most of the time. Normally until all hard constraints are satisfied the user does not interact with the optimization process.

**Iteration 18 (User Interaction):** At this point all hard constraints are satisfied and all objectives are within limits, as is depicted also in Fig. 9. Moreover, Fig. 10 confirms that the functional soft constraint meets the demand. Even though for a small period of time it goes below the good curve, it is considered satisfied because it is specified as a soft constraint. Since all constraints are satisfied and the objectives are within limits we have a valid, feasible design. If the user is satisfied, the optimization can stop here. If we continue the process without any changes the subsequent iterations will give also feasible designs, due to the FSQP solver used by Consol-Optcad. We observe that to meet the power demand at this stage, a lot more power is spent than needed. Therefore, we interact with Consol-Optcad and make the limits for fuel cost and emissions tighter and lower the power output, to force the optimizer to find a solution that will be more efficient while keep satisfying all constraints and objectives.

**Iteration 95 (Final Solution):** At the 95<sup>th</sup> iteration, we get a design that satisfies all hard and soft constraints and also meets the new tighter limits for fuel cost and emissions (Fig. 11). Fig. 12 confirms that the demand is met and because of the tighter limits on objectives, less power is needed with the current design to achieve the desired result.

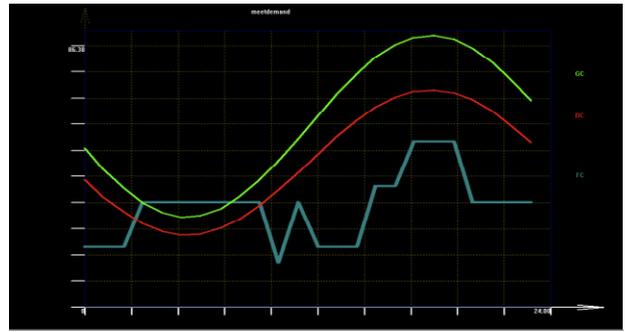


Fig. 8. Functional constraint after first iteration

Performance Comb (Iter= 18) (IPhase 2) (MAX\_COST\_SOFT= 0.851323)

Type	Name	Present	Good	Performance Comb	Bad
●	Con1 timeli...	1.200e+001	3.000e+000	<----- ----- ...	1.000e+000
●	Con2 timeli...	4.163e+000	3.000e+000	*----- ----- ...	1.000e+000
●	Con3 timeli...	8.000e+000	4.000e+000	<----- ----- ...	2.000e+000
●	Con4 timeli...	5.500e+000	2.000e+000	<----- ----- ...	1.000e+000
●	Con5 timeli...	7.837e+000	2.000e+000	<----- ----- ...	5.000e-001
●	Con6 timeli...	4.398e+000	2.000e+000	<----- ----- ...	5.000e-001
●	Con7 timeli...	6.744e+000	5.000e+000	*----- ----- ...	2.000e+000
●	Con8 timeli...	6.500e+000	4.000e+000	<----- ----- ...	2.000e+000
●	Con9 timeli...	6.744e+000	5.000e+000	*----- ----- ...	2.000e+000
●	F... meetde...	4.028e+001	4.855e+001	----- -----	3.884e+001
●	Obj1 fuelcost	7.132e+002	5.000e+002	===== =====*	1.500e+003
●	Obj2 emissions	1.249e+001	1.000e+001	===== =====*	1.800e+001
●	Obj3 operat...	3.433e-001	1.000e+000	====*	2.000e+000

Fig. 9. Pcomb after the 18<sup>th</sup> iteration

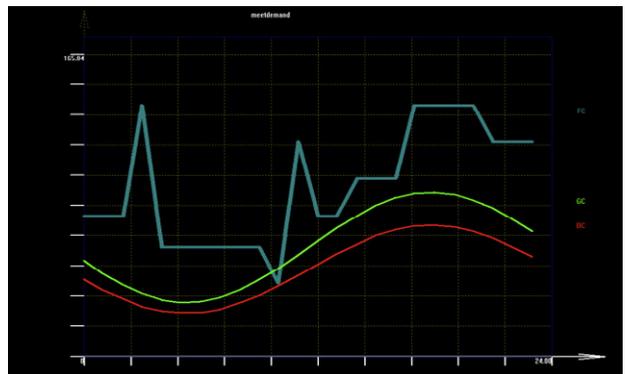


Fig. 10. Functional constraint after 18<sup>th</sup> iteration

Performance Comb (Iter= 98) (IPhase 2) (MAX\_COST\_SOFT= 0.997065)

Type	Name	Present	Good	Performance Comb	Bad
●	Con1 timeli...	1.200e+001	3.000e+000	<----- ----- ...	1.000e+000
●	Con2 timeli...	4.155e+000	3.000e+000	*----- ----- ...	1.000e+000
●	Con3 timeli...	7.214e+000	4.000e+000	<----- ----- ...	2.000e+000
●	Con4 timeli...	6.284e+000	2.000e+000	<----- ----- ...	1.000e+000
●	Con5 timeli...	7.841e+000	2.000e+000	<----- ----- ...	5.000e-001
●	Con6 timeli...	5.718e+000	2.000e+000	<----- ----- ...	5.000e-001
●	Con7 timeli...	5.202e+000	5.000e+000	*----- ----- ...	2.000e+000
●	Con8 timeli...	5.999e+000	4.000e+000	*----- ----- ...	2.000e+000
●	Con9 timeli...	6.709e+000	5.000e+000	*----- ----- ...	2.000e+000
●	F... meetde...	3.898e+001	4.855e+001	----- -----	3.884e+001
●	Obj1 fuelcost	5.710e+002	3.500e+002	===== =====*	6.500e+002
●	Obj2 emissions	1.099e+001	8.000e+000	===== =====*	1.100e+001
●	Obj3 operat...	3.285e-001	1.000e+000	====*	2.000e+000

Fig. 11. Pcomb after the 95<sup>th</sup> iteration

7. Conclusions and Future Work

In this paper we presented the modeling hub as a way to realize the Model-Based Systems Engineering vision and face today's challenges on systems synthesis and development. Furthermore, we focused on the trade-off path of the proposed hub and proposed a framework for integrating SysML with Consol-Optcad. The paper provided details on how each step of the integration was implemented and what tools were used throughout this process. The SysML Consol-Optcad

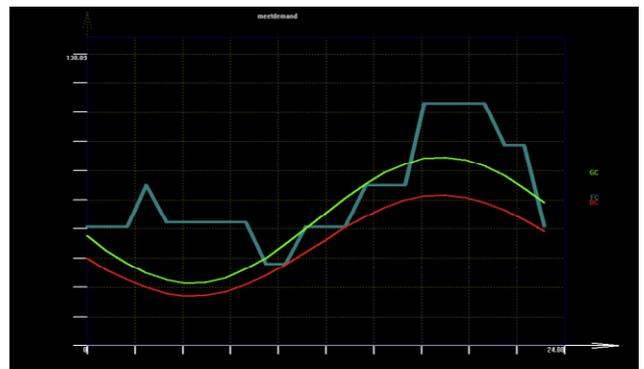


Fig. 12. Functional constraint after 95<sup>th</sup> iteration

The paper provided details on how each step of the integration was implemented and what tools were used throughout this process. The SysML Consol-Optcad

integration facilitates the problem formulation for the user and also enables the design and optimization processes, interacting and working in parallel in order to achieve the best possible design. A trade-off problem for an electrical microgrid was developed and solved to demonstrate the utility of the integration.

In the future we intend to expand the capabilities of this integration by making Consol-Optcad able to handle mix integer problems, which represent the majority of problems that industry usually faces. Making a two-way transformation is also planned for the near future, as it will give the capability of a true interaction between SysML and Consol-Optcad without human intervention. Finding a way to incorporate structural changes to the design space exploration process is another very challenging task that can expand the usefulness of the integration. Finally, as mentioned also in section 2, we plan to integrate IBM CPLEX and IBM-ILOG Solver in our modeling hub -- tools that are used widely in industry with excellent results in many domains.

## Acknowledgment

Research supported in part by the NSF under grant CNS-1035655, by the NIST under contract 70NANB11H148.

## References

1. S. Friedenthal, A. Moore, R. Steiner, *A Practical Guide to SysML*, The MK/OMG Press, 2009.
2. No Magic, Inc., *Paramagic-User Guide*, Version 17.0.
3. M. Bajaj, D. Zwemer, R. Peak, A. Phung, A. Scott, M. Wilson, *Satellites to Supply Chains, Energy to Finance-SLIM for Model-Based Systems Engineering Part 1: Motivation and Concept of SLIM*, Proc. INCOSE International Symposium, 20-23 June 2011, Denver CO, U.S.A.
4. B. I. Min, A. A. Kerzhner, C. J. J. Paredis, *Process Integration and Design Optimization For Model-Based Systems Engineering with SysML*, Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, August 29-31, 2011, Washington, DC, USA.
5. International Council on Systems Engineering (INCOSE), *Systems Engineering Vision 2020*, Version 2.03, TP-2004-004-02, 2007.
6. J.S. Baras, Lecture Notes for MSSE class, ENSE 621, 2002.
7. C. Haskins, K. Forsberg, M. Krueger, D. Walden, D. Hamelin, *Systems Engineering Handbook*, INCOSE, San Diego, CA, 2011.
8. J. Meyer, M. Ball, J. Baras, A. Chowdhury, E. Lin, D. Nau, R. Rajamani, and V. Trichur, *Process Planning in Microwave Module Production*, Proceedings SIGMAN: AI and Manufacturing: State of the Art and State of Practice, pp. 120-126, Aug. 31-Sept. 2, 1998.
9. M. K.H. Fan, A. L. Tits, J. Zhou, L.-S. Wang and J. Koninckx, *CONSOLE-User's Manual*, Techn. Report, Un. of Maryland, Vers. 1.1, 1990.
10. M. K.H. Fan, L.-S. Wang, J. Koninckx and A. L. Tits, *Software Package for Optimization-Based Design with User-Supplied Simulators*, IEEE Control Systems Magazine, Volume 9, Issue 1, Pages 66 - 71, January 1989.
11. M.B. Tischler, J.D. Colbourne, M.R. Morel, D.J. Biezad, *A Multidisciplinary Flight Control Development Environment and its Application to a Helicopter*, Control Systems, IEEE Journal, Volume 19, Issue 4, Pages 22-33, August 1999.
12. P.J. Potter, *Parametrically Optimal Control for the UH-60A (Black Hawk) Rotorcraft in Forward Flight*, MS Thesis, Un. of Maryland, 1995.
13. No Magic, Inc., *UML Profiling and DSL-User Guide*, Version 17.0, 2011.
14. The eMoflon team, *An Introduction to Metamodelling and Graph Transformations with eMoflon*, V 1.4, TU Darmstadt, September 2011.
15. A. Anjorin, M. Lauder, S. Patzina, A. Schrr, *eMoflon: Leveraging EMF and Professional CASE Tools*, INFORMATIK '11, Bonn, Oct. 2011.
16. T. Fischer and J. Niere and L. Torunski and A. Zundorf, *Story Diagrams: A New Graph Grammar Language based on the Unified Modeling Language and Java*, Universitat Paderborn, 2000.
17. No Magic, Inc., *Open API-User Guide*, Version 17.0.1, 2011.
18. A. K. Basua, S.P. Chowdhuryb, S. Chowdhuryb, S. Paul, *Microgrids: Energy management by strategic deployment of DERs - A comprehensive survey*, Renewable and Sustainable Energy Reviews, Volume 15, Issue 9, Pages 4348-4356, December 2011.
19. F.A. Mohamed, H.N. Koivo, *Power management strategy for solving power dispatch problems in MicroGrid for residential applications*, IEEE International Energy Conference and Exhibition (EnergyCon), Manama, Bahrain, 18-22 December 2010.
20. S. Jantti, *Connection of Distributed Energy Generation Units in the Distribution Network and Grid*, CODGUNet Project Final Report, Merinova Technology Center, Vaasa, Sweden, 2003.
21. W. Morgantown, *Emission Rates for New DG Technologies*, Regulatory Assistance Project, 2001.
22. H. Vahedi, R. Noroozian, S.H. Hosseini, *Optimal Management of MicroGrid Using Differential Evolution Approach*, 7th International Conference on the European Energy Market (EEM), Madrid, Spain, 23-25 June 2010.
23. NAHB Research Center, Inc., *Review of Residential Electrical Energy Use Data*, Upper Malboro, Maryland, USA, 2001.