Architecting Systems Under Uncertainty with Object-Process Networks

Benjamin Koo[1]
A-P Hurd[1]
David Loda[1]
Dov Dori[1,2]
Edward F. Crawley[1]
[1]Engineering Systems Division
Massachusetts Institute of Technology
[2]Information Systems Engineering
Technion, Israel Institute of Technology, Haifa, Israel
{bkoo, aphurd, dloda, dori, crawley}@mit.edu

Abstract

Architects of complex systems and products, such as an aircraft power system must routinely make tradeoff decisions given limited resources, incomplete information, and evolving stakeholder needs. Their decisions ultimately require a certain amount of subjective judgment, as a result of which system architecting involves not just science and engineering, but also art. By embedding Bayesian belief propagation algorithms in Object-Process Methodology (OPM), we show that architects of large-scale complex systems can better describe, quantify and communicate system attribute tradeoffs under uncertainty. This new framework, Object-Process Network (OPN), is instrumental in formulating and preserving the rationale behind tradeoff decisions, thereby offering a systematic approach to system-related decision making. Object Process Networks achieve the following objectives:

1. Graphically compose component-level knowledge to help visualize complex interrelationships among variables in a system,
2. Model the space of architectural options constrained by deterministic rules or conditional probability functions between system variables, and
3. Compare and select preferred architectural options under uncertainty.

To demonstrate the use of OPN in complex system architecting, we have applied this framework to an architectural tradeoff case study of an aircraft power system. A simplified architectural OPN model and its model construction process are presented and discussed in this paper.

Introduction

The architectures of large-scale engineering systems are derived from the socio-technical interactions of a wide range of stakeholders. Resolving resource contentions and identifying the degree of concession that needs to be made by all the sides requires open communication channel between stakeholders. As Weaver [Weaver 1949] pointed out, a coordinated system requires a language that is not only precise in its representation, but also concise enough to be managed by individual stakeholders. We are not aware of any effective system for engineers, scientists, managers, and budget controllers to communicate and negotiate their wants and needs during the system architecting process.

Product architecting can be described as a process of interactive knowledge discovery [Latour 1998] from which architectural decisions, driven also by stakeholders' interactions, often emerge. Each architectural proposal is likely to trigger a wide range of reactions and actions from different stakeholders. Rather than focusing on the efficiency of certain optimization algorithms, architects need an instrument to assess and explore the interactive effects across a wide range of variables. However, the amount of interactive scenarios in a real-world product development project dwarfs what most close-formed analytical techniques can generate and test. To meet this challenge, architects of engineering systems need a language that can incorporate technical facts and can present the facts in a meaningful fashion to various stakeholders involved in the architecting process. In practical terms, this means:

1. Representing system in terms of well-defined variables and relationships that encode the combinatorial solution space,
2. Constraining and testing the solution space to assess the feasibility and utility of different architectures, and
3. Providing a platform for problem definition that allows multiple users to specify and constrain the problem in a decentralized way.

Making it easy for all stakeholders to read and construct a network of variables and relationships is critical to allow for continued communication about concept and design up and across the organization, providing late-stage opportunities to avoid local optima. The system we have developed to meet the requirements above expresses relationships among system components and calculates the utility of architectural alternatives. It incorporates Bayesian Belief Network (BBN) [Pearl 2000] into Object-Process Methodology (OPM) [Dori 2002], a bimodal graphical-textual system specification paradigm that expresses the system's structure and behavior in a single model. The combination of BBN and OPM allows multiple participants to build a system model using graphical abstractions while retaining the qualitative and quantitative information regarding variable interactions. Our system also allows for continual concept testing, even as the design is refined. While product and organization architectures may become fixed to enable detailed design and development, the architecture of the accompanying information system stays flexible, a feature that may be particularly advantageous in projects with high-degree of complexity, such as aircraft power system development.

The OPN Framework

Graphical languages are most suited for mapping the interrelationships between a number of variables or attributes in a system. In particular, graphical modeling languages such as UML are designed for large-scale system modeling. However, their respective limitations prevent wide adoption or rigorous deployment. For example, UML is essentially a combination of multiple graphical languages where synchronization of evolving views in different languages (diagram types) is done manually. This multi-language approach creates additional overhead in terms of modeling synchronization, user learning, and communication-related efforts [Peleg & Dori 2000], introducing additional unnecessary complexity on top of the already complicated representation of a complex system [Dori 2002A]. To circumvent these problems, OPN is used as a basis for our modeling effort.

Executable Graphical Language

OPN is an executable graphical language based on the graphical notion of Object-Process Methodology (OPM) [Dori 2002]. OPM subsumes several modeling language formalisms, including entity-relationship diagrams, data flow diagrams, state transition diagrams, and a constrained subset of English, into a coherent, concise set of graphical/textual primitives. For reasoning under uncertainty, OPN extends OPM by encoding knowledge about variable interactions using a set of conditional probability functions. From a graph-theoretic viewpoint, OPN is a directed bipartite graph. An OPN “Object” represents a physical object or an attribute of that object, namely one of the object's variables with a range or a discrete set of possible states. An OPN “Process” represents a physical process that transforms an object or a decision driven by a conditional probability function. The directed links encode the direction of dependency between variables. For users to observe the interactions between variables, the state of a variable is displayed in as a round-edged rectangle inside the object. For discrete variables, a set of round-edged rectangles represents the possible values of the corresponding variable. The marginal probability of each variable is displayed as a bar-chart immediately adjacent to the corresponding objects and states.
OPN is made executable by incorporating a belief propagation algorithm (BPA) with a discrete event simulation engine to automate the reasoning procedure for variable interactions. The BPA we employed here is known as the Variable Elimination Algorithm, originally implemented by Cozman [Cozman 2000]. In this work, we implemented the algorithm only to infer marginal probability functions for discrete variables. In principle, our method applies to both continuous and discrete variables. Detailed description of various classes and computation properties of BPA can be found in [Murphy 2002].

Example: Aircraft Power System Architecture

To illustrate the utility of OPN in architecture tradeoff study, we will examine two aircraft power system architecture options, electrical and bleed-air driven. In a

traditional airliner, a small turbine engine, called the Auxiliary Power Unit (APU). It is located in the tail and is used to provide high-pressure air bled from its compressor (bleed air) to pneumatic starters on each main engine for start. This same high temperature air is also piped to the Environmental Control System (ECS) to provide heat for the cabin. Once the main engines are on line, the APU is turned off and the bleed air source for cabin heating and pressurization comes from the main engines for the duration of the flight. Bleed air also provides other subsystem functions, such as wing and cockpit window deicing and pressurization of fuel systems. These subsystem relationships and interactions dictate the design specifications required for each component. Since bleed air from the main engines is required to run other subsystems, fuel efficiency and thrust are impacted, affecting engine size and consequently the performance of the aircraft as a whole.

In the 7E7 design, Boeing engineers made the architectural decision to move to electric power as a means of reducing complexity and weight, with the intent of creating a more efficient and maintainable aircraft. The air in the cabin would be heated by electric elements, and the engines would be started by electric starter-generators. The impact on the design of the APU, engines and heating system is immediate. First, by significantly reducing the requirement for subsystem support bleed air, the engines can operate more efficiently from a thrust and fuel economy perspective, as more compressor air can be used for combustion. Second, the electrically heated ECS and engine starters would require a larger generator capacity. Hence, the APU design will be impacted, as it will now need to produce enough torque to turn a larger generator, but alternatively will not have to provide much in the way of bleed air.

These qualitative statements and anecdotal knowledge must be translated into quantitative measures and put in a holistic context to reason about their effects on the system. Moreover, decision-makers must assess the interactions between such anecdotal knowledge in an aggregate and quantitative fashion in order to make an executive judiciary architectural decisions. Qualitative statements can be encoded into a formal language to assist architects' systematic reasoning and decision-making. However, even with a small number of variables, the full factorial combinatorial possibilities can grow exponentially. This renders the analytical task of modeling a system that involves a large number of variables humanly and computationally intractable.

Utilizing available domain knowledge or statistical evidence, one can use OPN to specify variable dependency information, henceforth help architects to focus on a subset of variable interactions rather than assuming full factorial interactions between variables. A network-based modeling language like OPN allows architects and domain experts to incrementally model the direct interactions between a small set of variables in the system, and then infer the indirect interactions through network links. Through our OPN tool, changes in any part of the model will propagate throughout the network, enabling architects to reason about emergent properties of variable interactions. As the number of variables in a system grows, architects are not likely to be interested in all the variables in the system. OPN leverages the Markov condition

in BBN, which assumes that variables only interact with variables that are directly connected with them. This property helps architects to quickly construct a model of variable interactions by constructing a network of conditional probability functions that ultimately interact with all variables through the network links. This enables architects to focus on a subset of all possible variable interaction scenarios while considering subsystems and the systems' environment.

Modeling the space of architectural options

The model of the Aircraft Power System is shown in Figure 1.

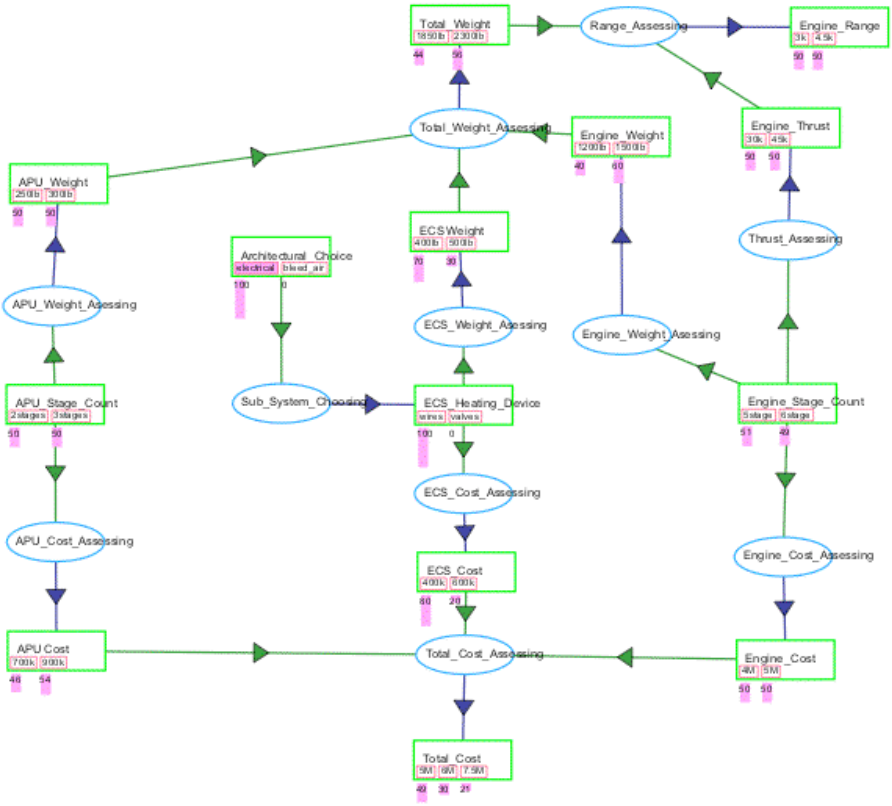

Figure 1. A model of the Aircraft Power System architectural options in OPN

In this OPN, our system is abstractly represented by a set of variables that can be qualitative (e.g., "electrical" vs. "bleed_air" architectural choice) and quantitative ("Engine_Weight" or "APU_Stage_Count"). For both these variable types, conditional probability functions are consistently used to encode variable interactions. In Figure 2, we show how a dependency is encoded as a conditional probability function. Using conditional probability functions not only captures the intuitive

knowledge of variable interactions, it is also an adequate mathematical representation of uncertainty, which can be adjusted based on statistical observation [Liu 1998]. Figure 2 illustrates the dependency between "ECS_Heating_Device", and "ECS _Weight", along with its cascading effect through the model, where "Engine_Weight" and ultimately "Architecture_Choice" are determined (see Figure 1.). The direction of the arrows indicates how conditional probability table is constructed. The belief propagation algorithms traverse the entire network by following the dependency links bi-directionally and use the conditional probability tables embedded in the process to calculate the marginal probabilities for all variables in the network.

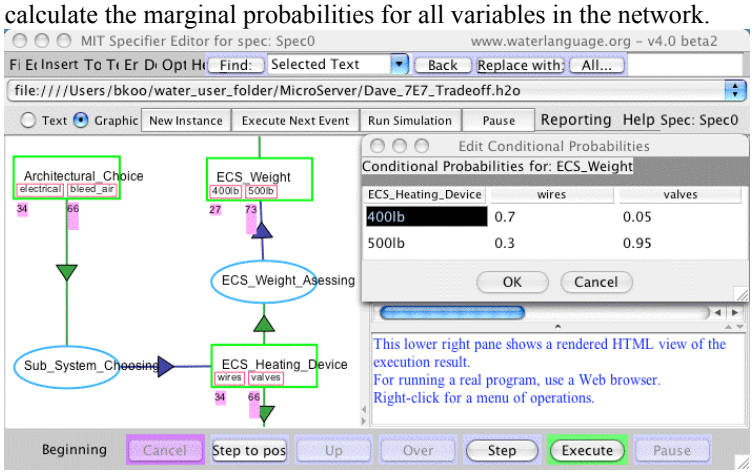

Figure 2. Part of the OPN diagram with a conditional probability table encoding the relationships between ECS_Heating_Device and ECS_Weight as a function of the Architectural Choice

As the architect choose "wires" or "valves" for the "ECS_Heating_Device", the marginal probabilities (the bar charts for each "Object" in Figure 2.) of "ECS_Weight", and "Architectural_Choice" will be updated according to the expert-specified conditional probability functions. A change in the marginal probability of any variable automatically updates the marginal probabilities of the rest of the variable. This provides the architects with a quantitative assessment of the effect of a local change across the entire system.

Composing system-level knowledge from component level knowledge

Clearly, knowledge about conditional probability functions is essential to evaluating architectural tradeoffs with BBN. While this type of probabilistic data is not always available [Lerat 2003], large complex systems tend to reuse components about which statistical information has been historically captured. In cases where direct relationships or detailed statistical data is not available, expert knowledge can often provide a reasonable assessment of variable relationships. As noted, the OPN software enables users to interactively assess the effect of any variable changes on all other variables. This facilitates the discovery of indirect relationships as the process

of architecting proceeds. The instant feedback regarding system variable interactions helps architects to systematically visualize the system as a whole, rather than focusing on individual attributes or performance metrics, which often results in sub-optimal decisions.

Making architectural decisions

Optimizing for the highest performance metrics is not the only objective in architectural decision-making. Architects must compare other conditional assumptions between different architectural options. Therefore, it is necessary to keep track of different conditional assumptions in each architectural assessment report. OPN's calculation engine can help architects obtain the expected values of performance metrics while presenting the marginal probability distribution functions of all the variables in the system model. Together, the marginal probability distribution functions and the dependency structures encoded in OPN provide the contextual knowledge about how the performance metrics can be attained. This holistic report of the system's global state help architects to ground their decisions with both qualitative and quantitative arguments. In this example, not only the "Total_Weight", and "Total_Cost" are used in the final decision, the cost of APU and ECS should also be taken into consideration, since different vendors might supply APU and ECS and their respective costs might have different supply chain implications.

Conclusions and Future Research

We have proposed OPN – a framework for jointly representing qualitative, quantitative and probabilistic information about a system in order to formulate and optimize architectural decisions. This framework provides a computational formalism that helps migrate the reasoning process of complex system architecting from art to science. Second, we have demonstrated the feasibility of the OPN framework by augmenting OPM's graphical language (Object-Process Diagrams) and implementing a prototypical language interpreter that supports Bayesian Belief Networks. This unified representation of complex systems reduces the cost of modeling and stakeholder communication errors. More importantly, it demonstrates that once a correct conceptual model has been constructed at an abstract level, laborious combinatorial and probabilistic reasoning can be carried out mechanically. Once a change is made in any part of the graphical model, the effect of change is instantly propagated over the rest of the model. This is an effective communication mechanism that makes system-level emergent properties visible to architects. Using probabilistic measures, the degree of impact can be calculated. Decisions or comparisons can be made analytically or numerically, depending on the resolution of the model.

OPN will help stakeholders visualize and elevate the quality and scale of architectural choices in order to optimize architectures of complex systems. This systematic knowledge representation tool will ground the negotiation processes amongst multiple

stakeholders, reducing confusion and premature optimization in complex architectural decisions. Our experimental OPN tool is designed to support these knowledge management concepts. The tool can be extended to incorporate also model inputs and updates directly by users over the Internet or from direct continuous information sources. Ultimately, treating executable system representations on the web in XML format would provide a scalable knowledge management infrastructure. Our OPN implementation leverages an executable XML technology to reap the benefits of the scalability of existing web infrastructures.[Fry 2002].

Acknowledgement

The authors would like to thank Christopher Fry and Michael Plusch for their implementation of this prototypical OPN modeling tool. We also would like to thank Dr. Geilson Loureiro and Jay Conne for their suggestions and corrections in this paper.

References

Shannon, C., & Weaver, W., 1949, *The Mathematical Theory of Communication*, University of Illinois Press (Chicago).

Latour, B., 1998, *Science in Action: How to Follow Scientists and Engineers Through Society*, Harvard University Press (Cambridge).

Pearl, J., 2000, *Causality: Models, Reasoning, and Inference*, Cambridge University Press (London).

Dori, D., ,2002, *Object-Process Methodology*, Springer-Verlag (Berlin).

Dori, D., 2002A. Why Significant Change in UML is Unlikely. *Communications of the ACM,* Nov., pp. 82-85.

Peleg, M, Dori, D., 2000, *The Model Multiplicity Problem: Experimenting with Real Time Specification Methods*, IEEE Transaction on Software Engineering Vol. 25, No. 8, August 2000.

Cozman, G.F., 2000, *Generalizing Variable Elimination in Bayesian Networks*, Workshop on Probabilistic Reasoning in Artificial Intelligence, November, 2000., Atibaia, Brazil

Cozman, G.F., 2001, *JavaBayes version 0.346*, online electronic media, http://www-2.cs.cmu.edu/~javabayes/Home/ , last accessed: April 4, 2004.

Lerat, M.J., 2003, *Four Applications of Bayesian Network for Systems Engineers*, INCOSE 2003 13th Annual International Symposium Proceeding, p.1407-1422.

S. Zilberstein and S. J. Russell. In S. Natarajan (Ed.), 1995, *Imprecise and Approximate Computation*, Kluwer Academic Publishers.

Liu C.L., Wellman, M.P., 1998, *Incremental tradeoff resolution in qualitative probabilistic networks*. Fourteenth Conference on Uncertainty in Artificial Intelligence, July 1998.

Murphy P. K., 2002, *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. Thesis in Computer Science, University of California, Berkeley, Fall 2002.

Fry, C., Plusch, M., Lieberman, H., 2002 *Spinning the Semantic Web* , Dieter Fensel, James Hendler and Henry Lieberman, eds., MIT Press (Cambridge).