# Systems of Systems Architecting and Integration; Visualizing Dynamic Behavior and Qualities

Gerrit Muller
Systems Engineering
University of South-Eastern Norway
Kongsberg, Norway
gerrit.muller@usn.no

Kristin Falk
Systems Engineering
University of South-Eastern Norway
Kongsberg, Norway
kristin.falk@usn.no

Elisabet Syverud
Systems Engineering
University of South-Eastern Norway
Kongsberg, Norway
elisabet.syverud@usn.no

This is a pre-copyedited version of a contribution published in 2019 14th Annual System of Systems Engineering Conference (SoSE) published by IEEE. The definitive authenticated version is available online at IEEE Explore: 10.1109/SYSOSE.2019.8753804.

***Abstract*—A major responsibility of architecting and integration is ensuring that desired dynamic behavior and desired qualities emerge from the interaction of components within the systems, between systems, and between the users and environment of the systems. A challenge is that organizational attention tends to be on the parts structure, which is determining organization, logistics, manufacturing, and servicing. At the same time, many developers lack the competence to capture dynamic behavior and the way qualities emerge.**

**This paper explores the role of Systems of Systems (SoS) architecting in relation to the impact on integration and looks at alternative ways to visualize dynamic behavior. Three different Systems of Systems, both Directed and Acknowledged SoS, are studied.**

**The visualizations in this paper aid in conceptualizing the system of interest to a larger stakeholder community. Visualizing dynamic behavior allows us to think and reason about potential changes and improvements of Systems of Systems. We encourage use of visualizing dynamic behavior from early conceptual phase and recommend that the developed visualizations are maintained and in active use throughout the SoS life cycle.**

***Keywords—Architecting, Integration, Conceptual Modeling, Visualization, Dynamic Behavior, Qualities***

## I. INTRODUCTION

Systems of Systems (SoS) are characterized by evolutionary and managerial independence. Emergence is an inherent result of systems integration. In fact, we design to get desired emergence and to prevent undesired emergence. The challenge is that increasing complexity makes it more difficult to foresee everything. The SoS paradigm complicates the integration further because we have to cope with independent evolution of constituent systems, with limited shared governance.

In the ideal world, Systems of Systems failures do not occur. Unfortunately, human creators of SoS are fallible, do not know everything, and do not foresee all emergence that happens when we operate systems in the field.

Systems of Systems development require interaction with a diverse stakeholder community. This paper looks at alternative methods to explore dynamic behavior in SoS. The purpose is to provide tools and methods that can be intuitive to diverse stakeholders outside or their domain. We are looking at representations which bridge over roles, domains and areas of functional expertise. However, the methods are not meant to replace Model Based System Engineering methods, nor the details provided using the SysML or UML.

Section II explores the role of Systems of Systems architecting in relation to the impact on integration and looks at alternative ways to visualize dynamic behavior. In Section III, we apply the techniques to three different Systems of Systems; both Directed and Acknowledged SoS are studied. Discussions and conclusions are found in Sections IV and V.

## II. ARCHITECTING: FROM PARTS TO QUALITIES

Systems of Systems are commonly divided into four main types: Directed, Acknowledged, Collaborative, and Virtual [1]. Only the Directed SoS is created to fulfil a specific purpose and has a central owner. All other SoS are centrally managed, ownership is at individual system level and the central governance is not necessarily a single party. SoS architecting is therefore not done by a single team [2].

The qualities of SoS architectures emerge from integration of independent systems. In this section, we look at how dynamic behavior can be conceptualized.

### A. Architecting and Integration

Architecting and integration are two sides of the same medal. Architecting provides guidelines for decomposition, function allocation, and allocation of contributions to qualities. Integration is making it work, and in this way, validating that architecting did not miss anything. Figure 1 shows a simplified V-model. The Figure positions architecting and integration respectively on the left- and right-hand side in the V-model, and shows the main deliveries of each area.

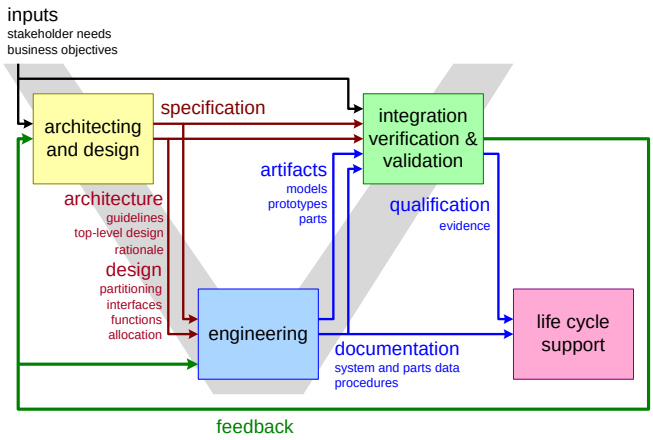


Fig. 1. Simplified Systems Engineering V-model

Conceptual modeling is part of "forward architecting". However, to be effective in systems integration, the system architect needs this conceptual understanding to step-wise confront the design with the real world.

The engineering and life cycle support functions, as well as the project organization have a strong focus on partitioning. The partitioning is the basis for the Work Breakdown Structure, and the Bill of Material, e.g. the backbone in the Enterprise Resource Planning systems that support life cycle functions as purchasing, manufacturing, and service.

### B. *From Parts to Qualities*

Dynamic behavior emerges through interaction of the parts, which should provide the functionality that the system promises. Figure 2 shows the parts at the basis. The dynamic behavior results in capabilities that the SoS delivers [3] [4].

Typical qualities are performance related, such as a) response time, throughput, and accuracy, b) trustworthiness related qualities such as safety, reliability, availability, security, and privacy, and c) economic qualities, such as total cost of ownership and cost per cycle. Bottom line, the qualities are the prime interest of the customer.

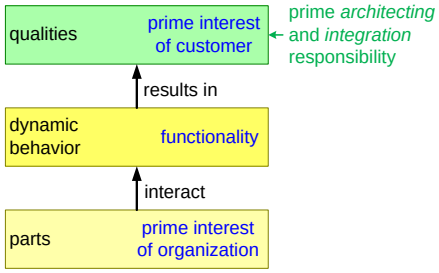


Fig. 2. From parts to qualities

The system qualities can be easier to observe for a single system, while the combined SoS behavior can be very difficult, sometimes impossible, to imagine. SoS qualities emerge from dynamic behavior of systems. Understanding dynamic behavior of the SoS is therefore essential to fulfil the customer needs.

### C. *Partitioning and Interfacing*

In Systems Architectures, the partitioning principle is applied recursively, creating a hierarchy of parts. Complementary to any partitioning are interfaces and every partitioning step introduces several interfaces. Interfaces [5] are a crucial means for managing engineering and life cycle functions. Partitioning stops when atomic parts are reached. Atomic parts are the leaves of the hierarchy. Typically, atomic parts are purchased and do not require further decomposition.

Systems integration requires early validation of the needs by modelling system behavior and qualities of the suggested architecture and iterating until the system has the desired behavior and qualities. We nowadays use many forms of hybrid system instantiations for feasibility studies and early validation and verification; these are steps in confronting the design with the real world. Examples are Hardware in the loop, Software in the loop, modified old systems, and simulations fed with real-word trace data. The design teams develop these hybrid instances during the detailed engineering phase and may capture emerging dynamic behavior later than desired.

### D. *Dynamic Behavior*

The interaction of parts over time results in an infinite amount of dynamic behavior. Even with a limited amount of parts, the amount of dynamic behavior is infinite; however, the System Domain and Systems Architecture usually constrain the dynamic behavior to the application envelope. This application envelope may change from one SoS to the next.

To explore dynamic interactions and understand the overall behaviors, the systems architect needs to approach the SoS from different viewpoints. Early involvement of stakeholders is important to avoid design changes and unplanned cost [6].

### E. *Technical Budgeting*

Architects need to allocate how functions and parts contribute to system qualities. A useful way to this is technical budgeting [3].

Figure 3 shows an example of a technical budget, the overlay (positioning accuracy) in a lithography system. Other examples of technical budgeting are center of gravity assessment, network sizing, and manufacturing tolerances.

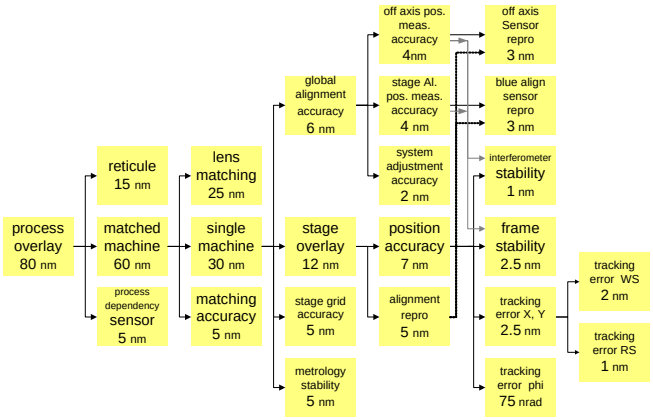


Fig. 3. Example of a technical budget, the overlay (positioning accuracy) in a lithography system.

### F. *Visualizations*

Visualizations are graphic representations of the system at various level of abstraction. Figure 4 shows a collection of visualizations of dynamic behavior briefly described below. This collection is far from complete. Its main purpose is to serve as inspiration for architects and designers.

The *Abstract Workflow* is a straightforward sequence of actions where the actions are abstracted by square blocks. The workflow displays only the order of the actions.

The *Cartoon Workflow* is a rather physical and concrete representation of the sequence of actions. Although the physical decomposition is still an abstraction and simplification of the real, the cartoon approach allows a better understanding of the physical relationship and interactions of the system in the workflow.

The *Timeline* shows the actions and adds the duration of the actions to the same workflow.

The T*imeline-and-Functional Flow* show a functional workflow where the functions are specified and relating each function to a timeline (when does this action take place).

*The Information Transformation Flow* shows how various process steps transform information from raw inputs into a world model for situational awareness. The tactical functions of the system use the world model to determine what the system should do next.

The *Information Centric Processing Diagram* is quite similar to the *Information Transformation Flow*. However, in this diagram the boxes are information. That means that the focus of this diagram is on the information, where the other diagram has the focus on the actions.

Physics oriented diagram and graphs here represented by the *Signal Waveforms* and the *Flow of Light* diagram captures how the physics property (e.g. magnetic field or light) travel with time or flows in the system and can also capture waveform shapes.

*State Diagrams* show the states of a system (or entity) and it shows when and how it transitions from one state to another state.

*Swim Lanes* are a common way to show concurrency. In this example, the swimming lanes represent physical functions in a metal printer. The horizontal axis in this example is a timeline. Sequence diagrams and interaction diagrams in UML use swimming lanes to show how objects interact; they often show the order, not time.

Additional graphs not shown in Figure 4 include, but are not limited to the following:

*Performance graphs* captures how the system qualities vary with changing conditions and can be effective in describing effect of variations in environment.

*Space Diagrams* are 2D or 3D representations of the physical space and can be effective in studying workflow and space constraints.

*Sequence Diagrams* shows how a system reacts with time throughout the sequence. The diagram can be efficient in capturing rate of changes.

*Flow Diagrams* represent flow into and from systems. In this example, power consumption visualized as incoming flow and the relative power consumption is visualized using thickness of the arrow. Also, note the relative heat generated by the components, which is visualized as flow out of the component.

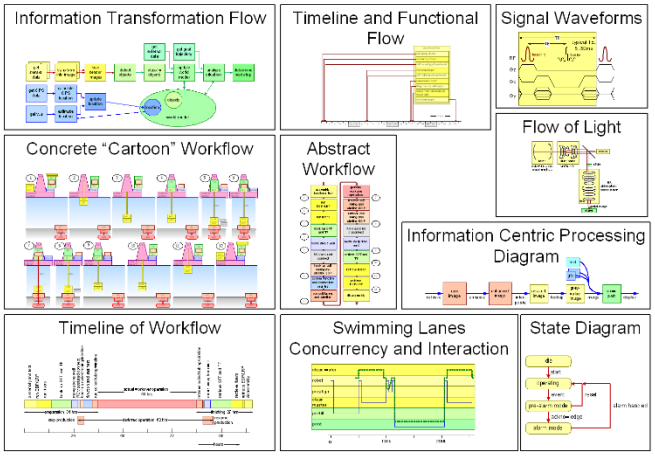


Fig. 4. Various examples of capturing dynamic behavior

## G. Capturing Dynamic Behavior

Visualizing dynamic behavior requires a conceptual mindset. When visualizing dynamic behavior, the focus needs to be on what happens, the functionality, rather than on the parts. Even in terms of what happens, the architects and integrators need a level of abstraction that allows them to see and understand the essence.

Another challenge of dynamic behavior is that it has many forms of appearance. A useful starting point for thinking about dynamic behavior is thinking in terms of Material, Energy and Information (MEI) flows [7].

When capturing dynamic behavior, time always plays a role, either absolute or event based. Dynamic behavior is about action and activities. In simplified form, it can be a single flow, however, in the real worlds there is often a large amount of concurrency. A trick to identify dynamic behavior is to look for verbs (functions or actions) rather than nouns (which are typically parts or components).

The analysis tends to start with typical circumstances and over time broaden to cover boundaries and exceptions of the application. Frequently, the real limits of the design are broader than what is covered in analyses, and sometimes the system designers will revisit and elaborate past studies to verify that the design can cover additional needs.

When capturing dynamic behavior of Systems of Systems, we recommend visualizing the most relevant dynamic behavior. Any attempt at completeness obfuscates insight and overview. A dilemma is that the devil is often in the detail. The challenge is to conceptualize (leaving out details, explaining it at a higher level of abstraction), while including the devilish detail.

In the next section, we apply the principles in three different case studies with two Directed SoS and one Acknowledged SoS.

# III. Case Studies

Visualization is a strong tool allowing early conceptualization of the overall behavior, which simplifies involvement from internal and external stakeholders. This is particularly useful for Systems of Systems because of the limited shared governance. In this section, we study three different cases of SoS where combinations of various visualization techniques help us identify challenges associated with integrating independent systems. The visualization can help clarify the associated challenge for multiple stakeholders.

## A. Workflow Disruption

Muller, Wee, Moberg [8] provide a case study of a workover operation in a Directed Systems of Systems. A workover operation is a maintenance operation of a subsea oil and gas well. This case study combines a physical model, several dynamic behavior models, and a quantification of the duration and cost of the operation on a single A3 Architectural Overview (A3AO) [9]. Figure 5 shows the complete A3. This visualization relates the dynamic behavior to the quality *duration and* explores the cost and the consequence of disruption.

The operation is shown as a series of actions in the abstract workflow, and the effect of each action on the physical model is shown in the cartoon workflow model. The cartoon

workflow improves the understanding of the relationship between the physical parts at each stage and allows a more intuitive understanding of the operation. The timeline shows the duration and order of the operations. The next information box, at the bottom-right, is a table that transforms the durations and resources used for each action into the cost. Adding all costs together results in the cost of the workover operation.

All models we discussed so far are "Happy flow" models, e.g. the operation when everything goes well. Reality is that there are many disruptions. The models in the middle of the A3AO show one example of a disrupted operation and its consequence on duration and order of the operations.

This A3 visualizes the entire line of reasoning from the parts in the physical model, via the dynamic behavior models to the duration to the cost of the workover operation and captures effects of disruption. By necessity, all models are simplifications. However, thanks to these simplifications, we have now a means to think about current performance and cost and ways to improve it.

### *B. Automation of Emergency System*

Broyde, Falk, Arntzen [10] provides a systematic study of 1171 reported incidents related to Emergency Quick Disconnect (EQD) systems installed with oil and gas drilling or production vessels equipped with dynamic positioning systems. This is a safety system in a Directed Systems of Systems. The system terminates the operation and disconnects the equipment if the operational envelope is outside of the design envelope. The authors studied two cases, a generic case and a specific case representing a real incident. The specific case occurred in 2004 and led to an EQD situation. This incident occurred during a deep-water drilling operation. Furthermore, environmental wind forces triggered the incident, and the secondary cause was an operator error. The published paper assessed the general problem using analytical and statistical representations.

Visualization of dynamic behavior allow for better understanding of the human behavior in the chain of events. Figure 6 shows three different views, a state diagram, a timeline diagram, and an information flow diagram.

The state diagram describes the behavior of systems and is commonly used in software development. The initial state is the Green Status or normal operations, and that is where the scenario starts. The state diagram is typically useful to describe the states in the system from normal behavior to failure to disconnect. Each state has a different alarm level and when the situation is handled, the system reverts to the previous state.

The middle diagram compare timeline of events for the two different cases. The Emergency Quick Disconnect is the final step of a chain of events. The timeline allows the reader to catch the chronology of a sequence of events (here the events leading to an EQD). Comparing the two timelines, the generic one that was based on procedures, and the specific one based on a case, we see clear differences.

The information flow is a functional model of information flow from the operator point of view. The visualization is similar to visualizations of other autonomous system. This is because the structure of autonomous systems tends towards an imitation of the already existing system architecture. The information flow diagram aids in identifying how to develop autonomous systems. The information flow diagram is used to reason about the operator active workload to detect situations known as operator "out-of-the-loop" where the operator is no more an active part of the process. Such situations should be considered for automation to enable early detection of unwanted situations.

### *C. Renewable Energy Inflow in Grids*

Electric grids are Acknowledged Systems of Systems where constituent systems retain their independent ownership and cooperative agreement governs changes. In such systems, the electric power market price is a driving force for operators of existing power plants. Increasing share of renewable energy production results in rising flexibility need in existing electric grids. Renewable energy produced by solar and wind varies with local meteorological conditions and can change significantly within minutes (e.g. when a cloud passes in front of the sun). To maintain grid stability, the existing power plants of the grid must compensate the variation in renewable energy inflow to the grid.

Figure 7 captures aspects of dynamic behavior that improve the understanding of the constraints of existing electric grid flexibility and the SoS capability to accept renewable energy without violating the electric quality requirements (voltage and frequency variations).

The combined time and functional flow diagram show the steps and associated time involved in a start/one-hour power production/stop cycle of conventional fossil fuel power plants (gas turbines and coal) based on data from literature [11]. The diagram allows the reader to appreciate the variation in transient timescale involved in various technologies and shows that these power plants do not necessarily have the flexibility required for frequent stop and restart cycles.

Overlaying typical renewable energy inflow variation with typical ramp time found in literature [11] in an actual timeline allows us to reason about the maximum capacity of an existing electric grid to accept renewable energy inflow. The example case use real time data from German power production in February 2019 [12] where the renewable power inflow is 33% at the start of day, increasing to a maximum of 42% at the peak and then falls to 21% when the sun sets. The maximum ramp rate is typical of an aero derivative gas turbine engine, or a large size coal plant. Seen in relation with the time and functional flow visualization only the aero derivative gas turbine has the required response time to compensate for the given solar power intermittency.

## IV. DISCUSSION

A challenge of Systems of Systems is that many independent systems interact. The challenge is to keep the understanding up-to-date, when constituent systems change independently. Visualization of dynamic behavior provides synthesis of knowledge captured through involvement from multiple teams and stakeholders from a given viewpoint.

In Directed SoS, dynamic behavior visualizations are made both by the central SoS owner as well as the individual System owner to capture constituent system behavior and effect on SoS. This is done during design phase from early pre-engineering through detailed design. It is of vital importance that the System owners have a global understanding of the SoS dynamic behavior to allow early detection of inconsistent behavior.

The dynamic behavior and emergent qualities can be complex to assess. This is particularly true with the more loosely organized Acknowledged SoS. Different stakeholders investigate such systems from multiple viewpoints. Unfortunately, this can lead to biased conclusions due to conflicting interests and fractured governance. Renewable inflow in electric grids serves as an example of such a topic. Visualizing dynamic behavior may aid in developing a better appreciation for the overall complexity through the combination of various viewpoints.

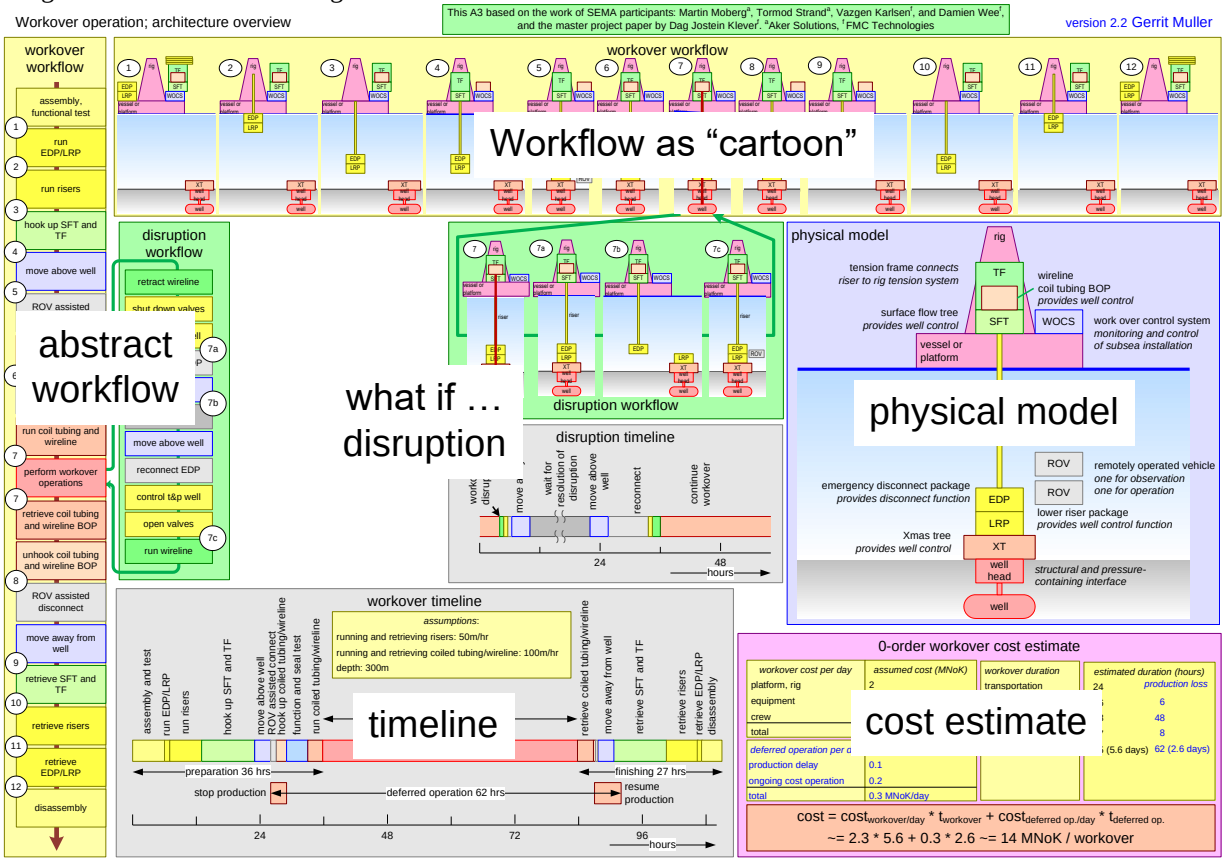


Fig. 5. The combination of physical, dynamic behavior model, contributions to qualities like duration and cost provide overview on one A3.

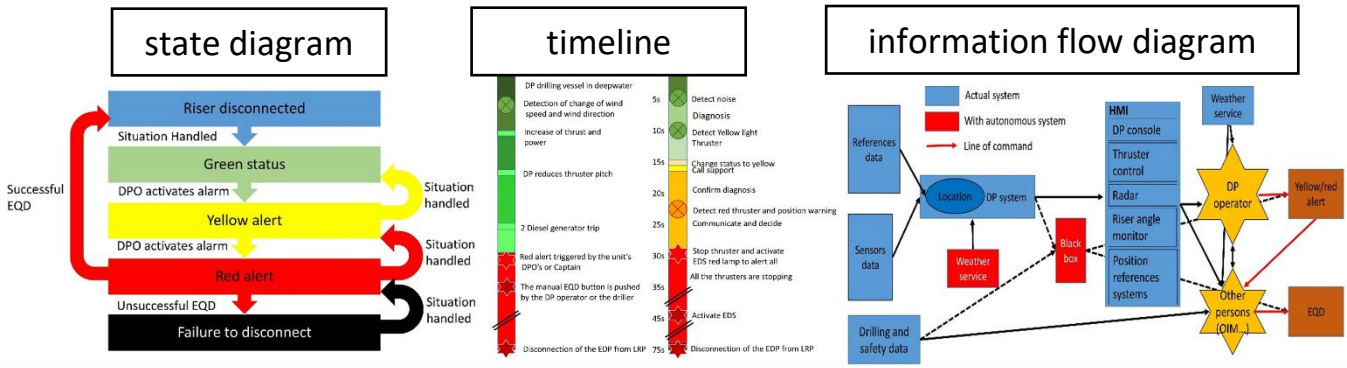


Fig. 6. State diagram, timeline and information flow diagrams used to visualize dynamic behavior in Emergency Disconnect System

information flow and timeline

timeline

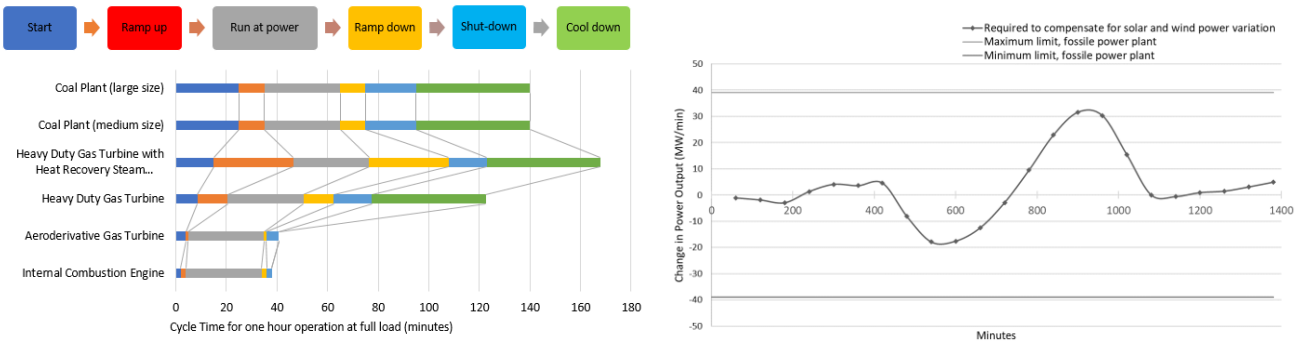


Fig. 7. Information flow and timeline diagrams for assessing dynamic behavior and operational constraints of electric power grid with renewable inflow

The visualization methods described in the paper are not designed primarily to help in the system integration phase, but to conceptualize dynamic behavior during the early design phase with the goal to verify the concept as early as possible. Consequences of late detection are high cost, schedule challenges and mitigation by operational constraints. Without dynamic behavior diagrams, we run the risk of large-scale trial and error.

The case studies presented in this paper show the importance of making visualizations to understand dynamic behavior and the way qualities emerge. Interestingly, when we show visualizations to engineers and managers from oil and gas companies the response was that they recognized that if they had done this beforehand, they would then have detected problems in specification and design at an earlier stage. Visualization techniques like the examples presented in this paper are found in engineering practice. Cross-functional teams use visualizations to aid in developing common understanding and reasoning. These models are often developed on the fly on the meeting room white board. However, these visualizations are rarely captured and kept for the future. Reports and minutes of meetings are commonly text based and the visualizations are often static and only supporting the text. We expect that combining visualizations into a common single view can be efficient in capturing static as well as dynamic behavior. Systematic use of visualizations has the potential to improve communication and documentation of the reasoning behind the architecting decisions.

Visualizing dynamic behavior can be challenging. To get started, we recommend exploring the system from the time perspective and look for potential disruptions and constraints emerging from interaction between systems and between system and operators.

## V. Conclusions

In this paper, we applied visualization principles to capture dynamic behavior in three Systems of Systems. Both Directed and Acknowledged SoS are studied. We found that visualizing dynamic behavior allows us to think and reason about potential changes and improvements of SoS.

Our case studies identified common challenges associated with integrating independent systems:

- Understand the impact on the workflow (how to handle) and the impact on time and cost with workflow disruptions.
- Compare impact on installation sequence and means for various concepts.
- Evaluate obstacles in information flow in emergency systems that interact with human beings.
- Understand technical constraints in existing technologies when interfacing new technologies.

Visualizations are a strong tool allowing early conceptualization of the overall behavior that simplify involvement from internal and external stakeholders. This is particularly useful for Systems of Systems because of the limited shared governance.

Multiple visualization techniques are available to assess Systems of Systems from different viewpoints. We recommend assessing behavior visually using alternative techniques.

This helps in reasoning across the multitude of stakeholders associated with SoS. We encourage use of visualizing dynamic behavior from early conceptual phase and recommend that the developed visualizations are maintained and kept as part of the document repository for active use throughout the SoS life cycle.

## VI. References


[1] Dahmann, J. and Baldwin, K. *Understanding the Current State of US Defense Systems of Systems and the Implications for Systems Engineering*, in IEEE Systems Conference, Montreal, 2008.

[2] INCOSE, 2007, *Systems Engineering Vision 2020*, INCOSE-TP-2004-004-02, from http://sebokwiki.org/wiki/INCOSE_Systems_Engineering_Vision_2020.

[3] G. M. Bonnema, *Funkey Architecting; An Integrated Approach to System Architecting Using Functions, Key Drivers and System Budgets*, Enschede: University of Twente, 2008.

[4] S. Haveman and G. M. Bonnema, *A conceptual model to support communication of systems modeling and simulation activities*, in IEEE-SysCon, Vancouver, 2015.

[5] D. M. Buede and W. D. Miller, *The Engineering Design of Systems: Models and Methods*, Hobkoen, NJ: John Wiley & Sons, 2016.

[6] Blanchard, B.S., and Fabrycky, W.J. *Systems Engineering and Analysis*, Upper Saddle River, NJ: Pearson Education, 2006.
[7] Hinsley, S.W., *Maintaining systems-of-systems fit-for-purpose: a technique exploiting material, energy and information source, sink and bearer analysis*, Loughborough: Loughborough University, 2017.
[8] Muller, G., Wee, D., and Moberg, M. *Creating an A3 Architecture Overview; a Case Study in SubSea Systems*, in *INCOSE*, Seattle, 2015.
[9] Borches, D. *A3 Architecture Overviews, A tool for effective communication in product evolution*, PhD thesis, University of Twente, 2010.
[10] Broyde, H., Falk. K., Arntzen, A.A.B, *Autonomous security system specification within oil and gas industry for offshore vessel: rewuirement and Concept*, Proceedings of SDPS, 22nd International Conference, Birmingham, AL, USA, (22:244-248), 2017.
[11] Gonzalez-Salazar, M.A, Kirsten T., Prchlik L., *Review of the oprational flexibility and emissions of gas-and coal-fired power plants in a future with growing renewables*, Renewable and Sustainable Energy Reviews, February 2018, Vol.82, pp.1497-1513.
[12] Data from European Energy Exchange, https://www.eex-transparency.com/power/, accessed 2019-Feb-13.