Valuing Investments in Agile Project Design: Example for Upstream Oil and Gas Development

by

Katherine A. Brown

B.S Mechanical Engineering
B.S. Industrial and Operations Engineering
University of Michigan, 2008

SUBMITTED TO THE SYSTEM DESIGN AND MANAGEMENT PROGRAM IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE IN ENGINEERING AND MANAGEMENT
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SEPTEMBER 2020

©2020 Katherine A. Brown. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Signature of Author: _____
System Design and Management
August 7, 2020

Certified by: _____
Richard de Neufville
Professor of Engineering Systems, Institute for Data, Systems, and Society
Thesis Supervisor

Accepted by: _____
Joan Rubin
Executive Director
System Design and Management Program

This page is intentionally left blank

Valuing Investments in Agile Project Design:
Example for Upstream Oil and Gas Development

by

Katherine Amae Brown

Submitted to the MIT System Design and Management Program on August 07, 2020
in partial fulfillment of the requirements for the degree of
Master of Science in Engineering and Management

Abstract

Traditional oil and gas companies will continue to face market uncertainties in the coming decades. With increased pressure to confront climate change and energy technology innovations, the future demand and pricing for petroleum products is unclear. Hydrocarbons will continue to play a part in the changing energy landscape. However, companies will need to revisit what capital is spent on and how they spend it. Instead of tying up capital in a one-time massive investment decision, made under long term assumptions, agile investing gives power back to the business decision maker.

This thesis has developed a computationally efficient model for valuing systems built for agile investing. It combines system architecture principles, real options valuation, and object-oriented programming. Investment decisions under uncertainty are simulated by combining optimization algorithms and Monte Carlo sampling. The approach allows expansion decisions to be included in the early stages of system architecture design. In industry, definition of subsystem requirements is an influential step in project development, setting up the costlier and time intensive detailed engineering, procurement and construction.

Practicality is demonstrated through application to a realistic, but hypothetical case study. We explore the development of an upstream, onshore oil field. The system is decomposed into several subsystems accomplishing fluid extraction, processing and sales. The model simulates their physical and economic interactions to calculate performance metrics of net present value, capital expenditure, system capacity, emissions and others. We investigate performance changes based on subsystem sizing and installation timing.

The analysis shows how agility can increase expected value while reducing investment risk. Overall expected value increases by 5% and the initial capital commitment is only one-sixth the cost of a full production system. The value is created by earlier positive cashflows, hedging commitment against falling oil prices and quick expansion opportunism in the case of rising prices. Using the same model, subsystems are refined and then expanded to investigate combustion emissions. By incorporating cleaner fuel sources, combustion emissions can be reduced by 70%. We conclude by recommending specific subsystem requirements for an agile investment design.

Keywords: agility, oil and gas, system architecture, real options, Monte Carlo simulation, integer optimization, managing uncertainty, design flexibility, object-oriented programming

Thesis Supervisor: Richard de Neufville
Title: Professor of Engineering Systems, Institute for Data, Systems, and Society

This page is intentionally left blank

Acknowledgements

The System Design and Management (SDM) program at MIT was a transformative year for me. I had not been full-time in a classroom since 2008. SDM was a year of reflection, growth and acknowledging how far I've come but how much I still have to learn. I admire and thank the SDM faculty for their courage. Facing a room full of seasoned professionals seems daunting. Their work towards bridging the gap between academic knowledge and industry application also so. I took Professor de Neufville's class and was intrigued by the concept of valuing design flexibility. He has a knack for explaining concepts simplistically and directly. Making them applicable to the real world, which I most appreciated as we worked together. Additionally, he was encouraging (and very patient) as I fumbled through tying in other concepts I had learned during the year and wanted to explore.

I want to thank my former supervisors and mentors who were my teachers and professors between my time in undergrad and SDM. Carlos Tovar was my first mentor on a large deepwater platform design team. He taught me the basics of upstream process, sizing of equipment and managing contractors. Nicola Killen taught me how to facilitate a technical conversation and tease out underlying knowledge from a large multidisciplinary group. She also sparked my international travel itch and confidence which would serve me years later. Ricardo Morales was my supervisor while I managed some of my favorite small capital projects. Always encouraging our team and trusting our decision making. He was a windshield from normal corporate politics and a door opener to my assignment abroad. Randy Potter was my advocate in one of my most challenging positions. He saw more in me than I saw in myself and kept moving the bar higher, so I kept being challenged. Andy Koch gave me lessons in leadership by creating environments for collaboration and communication. He encouraged and backed me when I was seeking management support to apply to SDM.

While coding and writing this work my grandmother passed away. Her and my grandfather paid for all of their grandchildren's undergraduate education. They were both automotive factory workers and never attended college. This investment was made when we were born and meant that growing up, I never questioned the value or my pursuit of higher education. It was just a fact, I was going to college. She taught me a lot of other life lessons, but my education success is also because of her.

This page is intentionally left blank

Table of Contents

List of Figures

List of Tables

List of Acronyms

BPD – Barrels Per Day

CNG –Compressed Natural Gas

CO_2e – Carbon Dioxide equivalent

DCF – Discount Cash Flow

GHG – Greenhouse Gas

GOR – Gas to Oil Ratio

INCOSE – International Council on Systems Engineering

IRR – Internal Rate of Return

LNG – Liquified Natural Gas

MBPD –Thousand Barrels Per Day

MCF – Thousand Cubic Feet

MMSCF– Million Standard Cubic Feet

MMSCFD – Million Standard Cubic Feet per Day

MW – Megawatt

MWh –Megawatt hours

NPV – Net Present Value

OOP – Object-Oriented Programming

OPD – Object-Process Diagram

OPM – Object-Process Methodology

NGL – Natural Gas Liquids

SysML – Systems Modeling Language

UI – User Interface

Chapter 1: Introduction

Uncertainty in the energy market

The world's energy system is complex and socio-technical. Its societal and technological elements interact in ways that are challenging to model, predict and change. Energy has been the essential and driving factor for the industrial revolution (Wrigley, 2010), an enabling component for further industry and technology advances (Stern & Kander, 2012) and unimaginable gains in standard of living. It lives on a global stage, integrated into politics, everyday life, the environment and economics. While governments, individuals or corporations may have direct economic benefits from supplying energy, there is no doubt that meeting the world's energy demand benefits the lives of every human.

Most renowned energy system models correlate the global population with the demand for energy, including that of the International Energy Association, ("World Energy Outlook 2019," 2019). As the world's energy use has increased, the pieces of this growing energy pie have not remained static. Decades of evolution and innovation in energy industries have changed the proportions and overall magnitude of the fuels used to meet demand: biomass, whale oil, coal, nuclear, biofuels, crude oil, natural gas, solar, wind and other energy forms. Oil and gas have highly efficient energy conversion ratios and are an abundant natural resource. This has created demand from downstream industries and has established the traditional oil and gas sector as the dominant fuel type. Oil and gas supplies over half ("World Energy Outlook 2019," 2019) of the world's energy needs.

Within the traditional oil and gas industry, new project investment, expansion and divestment are business decisions that are made regularly. Large scale and long-term investments have been typical in the past few decades (Ernst & Young, 2014). The cycle from discovery, full capital investment and finally to operations with positive cashflow can take a few years to over decades. The project's time frame varies depending on the technical complexity, the location's politics, local and global demand, and a firm's portfolio management strategy.

Conventional investment analysis evaluates spending capital infrastructure up front, with revenues and operation expenses coming in over the life of the asset. Historically, models show increasing projections in energy demand growth and fossil fuels remaining as the majority supplier. This has reinforced the behavior of energy companies investing in high capital, large scale, long lead projects. To take advantage of economies of scale they spend more upfront for larger projected gains later. Projects with this strategy can easily be communicated and value

computed with a discounted cash flow analysis. The assurance of demand creates a sense of certainty for revenues and coupled with an analysis that can easily be interpreted, large capital investment project strategies are endorsed and reinforced.

Oil and gas will likely continue to be the dominant fuel source in the next 20 years ("World Energy Outlook 2019," 2019). However, the magnitude of the overall energy demand and the role renewables will play has become increasingly uncertain. The IEA's 2019 World Energy Outlook examines the large gaps in the stated policy scenario and a sustainable development scenario, which meets the Paris Agreement. In the sustainable development scenario overall energy demand must decline and renewables account for a third of the supply, Figure 1. Regulation and policy, social drivers, consumer behavior and new technologies' cost and scalability will all play parts in how quickly and which solutions will be employed.

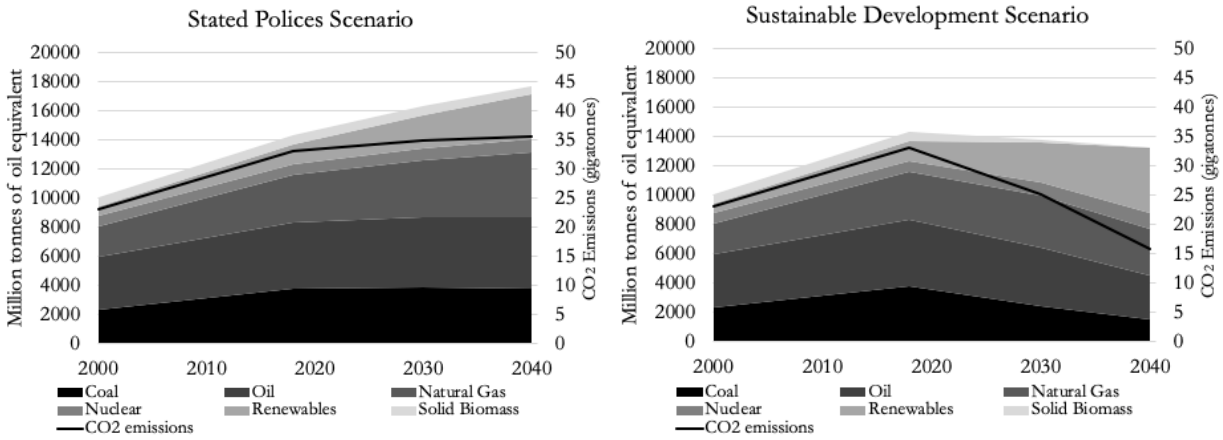

Figure 1: World primary energy demand by fuel and CO2 emissions for the "Stated Policies" and "Sustainable Development" Scenarios ("World Energy Outlook 2019," 2019)

The IEA states meeting the sustainable development scenario "would require significantly more ambitious policy action in favor of efficiency and clean energy technologies, including decarbonized fuels and a major rebalancing of investment flows." Through the report, government and policy makers have been shown that neither the existing nor the stated policies will achieve the Paris Agreement. The policies to be implemented in the future are uncertain, they may include additional carbon taxes, bans on fracking, further emissions reporting, etc.

The digital revolution has changed how society communicates and makes decisions. Global information is available quickly and from many sources. Layered on top of a highly informed society, empowered to make decisions with more information, is the topic of climate

change. Science and technology have proven and now convinced mainstream society that climate change is happening, and that human energy consumption has contributed to it.

Emphasis has been put on the energy system to slow climate change. The burning of fossil fuels has served society. However, end consumers have become increasingly aware of fossil fuel impacts and vocal about the need for solutions that do not have long term negative environmental impacts. Ideally solutions to the energy system would enable societal growth without negatively reinforcing other systems that society depends on (geography, weather, environment).

A significant uncertainty factor is technology development and innovation. More efficient technology in all energy sectors is continuously being created and tested. The technology breakthrough of fracking changed the United States' natural gas supply and demand system and significantly changed the country's role in the global energy system. Future technologies have the potential to be just as or even more trend breaking.

Leveraging and hedging uncertainty through design

Acknowledging uncertainty around policy action, social behavior and technology breakthroughs is the first step an energy company can take. The second step is leveraging it. Managing uncertainty is typically thought of as risk mitigation by minimizing downside risk, but it can also be used as an opportunity. A well-positioned firm ready to take advantage of the upside of demand, policy or technology uncertainty may have a competitive advantage over its peers.

"Readiness" can be a daunting challenge, there are many factors that play into a company being ready to benefit from trend-breaking and disruptive uncertainties. Factors such as workplace culture, available capital and organizational capabilities have had a long history of being studied in business schools and practiced at firms. There are also different strategies to address uncertainty: refined forecasting, diversification, robustness, agility and flexibility. Survival of a firm needs forecasting, diversification and robustness. Thriving in the face of uncertainty a firm needs the tools of agility and flexibility.

One way to introduce agility and flexibility into an energy producer's portfolio is through design. We can think of flexibility in design as the ability to accommodate changes to a physical system, while agility is the ability to react to these changes quickly. Investing and developing agile and flexible designs into oil and gas facility systems is done to some degree already. Flexible examples are adding a spare valve to allow for future pipeline tie-ins or designing the

inlet separation of a facility so that additional compressors can be added to expand capacity. These are small changes in flexibility, typically driven by operation requests for small expansions in production. Adding agility to the example we would use a standard size compressor, easily available and already designed so it can be installed quickly.

What if this idea was extrapolated further, if expansion agility was a significant driver in the overall system design and the projects that are invested in? Would the industry move away from investing in large, long lead, capital intensive projects in favor of agile, short cycle to production designs? Would different technologies, including those that perform better under policy action or changes in consumer behavior be deployed if evaluated based on future agility?

Moving away from large, long return capital investment projects could be a strategy to dampen risk from market uncertainties. Agile oil and gas asset development using smaller, faster return expansion projects could replace large major capital projects. Shortening the time from capital investment to positive cash flow is only one benefit of this strategy, the real benefit lies within the continual real option decisions. A real options designed project gives the business decision maker more levers at their disposal. They can continue or stop investment, reacting to more certain, near term conditions by making shorter cycle, smaller capital expansion decisions.

The value of providing a design that enables future decisions, sometimes called real options, can be challenging to quantify. It is much easier to predict a decision metric, such as net present value (NPV) if assumptions are made as deterministic inputs into a valuation model. Additionally, it is much easier to ignore potential future decisions based on future uncertainties. Therefore, simplifying models to evaluate designs where forecasts are assumed correct or only incorporating a few uncertainty possibilities using probabilities is typical practice. Then further "optimizing" the design under these assumptions occurs.

Using Monte Carlo simulation and stochastic modeling can give insights to decision makers on the value of investment agility within a project. Developing a model that includes forecast uncertainties, potential future actions and communicates the value created from expansion agility is not an easy task. It takes a mind shift from traditional assumptions in decision making, that long term forecasts are right and that decisions will not be taken in the future. A focus on the distribution of the potential outcomes and future decisions may be as important as the most likely value. (de Neufville & Scholtes, 2011).

Understanding the range of potential outcomes, allows companies to plan for the less likely scenarios. They can mitigate the worst scenarios or take advantage of best-case conditions.

For example, knowing an investment may lead to significant losses can lead to creative solutions (like agile, staged development) to minimize capital losses. They can design the system where the real option to stop and divert investment still takes advantage of what has been invested and minimizes capital loss. On the other hand, knowing that there is potential for great upside if conditions turn favorable, the system can be designed for easier and quick expansion.

Learning from the digital industry

The digital revolution has supported more uncertainty in social behavior towards energy, but at the same time it can be leveraged to support a change in decision making. Digital technologies have not only created an acceleration in information sharing but also decision making. More specifically, the exponential growth in computing power, programming capabilities and mainstream adoption of digital technologies has enabled support tools used for decision analysis. Using modern digital technologies, computation and information manipulation can now be done faster and at a scale far beyond what was capable just a decade ago. It's not just the data that has created this step change in decision making power. The power to compute and skills to consolidate and visualize available information is also enabling better decisions.

One could argue that the agile project management process used in software development is a manifestation of rapid real option deployment. By incrementally releasing useable product updates the methodology accelerates the end user feedback loop. The work is developed in a short "sprint cycle" of typically two weeks with a small, dedicated team. The product is released incrementally after each sprint to the customer. The product reception is uncertain but quickly evaluated by the user and then integrated into the next round of product development. The approach enables or defers future decisions in the product, allowing agile project managers to create successful products in an uncertain world. (Racheva & Daneva, 2010).

In addition to proving out value in agility, the software industry has also increased the computing power and software tools that can be used in model development. A Monte Carlo simulation can be done quickly with many variables of uncertainty. Digital technology enables these models to run faster and grow bigger.

Simulating investment agility of a complex system

An agile investment strategy for project development can be described qualitatively, making logical sense and leveraging examples from other industries. The challenge with creating a fully convincing argument is a quantitative comparison to a traditional investment project. For

a one-time investment or one with a few future decisions, a cash flow or decision tree analysis, can be done. These analyses can bias towards large investments that leverage economies of scale.

In contrast, a project with investment agility gives the decision maker opportunity to make or not make many investment decisions continually, as conditions of the market and the system itself changes. Given certain conditions, the aggregation of these smaller investments could amount to a larger expenditure than the large-scale project investment. Yet under uncertain conditions the choice to not spend (and invest the money elsewhere) or spend more (and gain greater profits) may outweigh the cost of smaller scale.

Investing in a system designed to be agile is becoming even more relevant for energy companies who want to thrive in the uncertainty of the next decades. Quantifying the value of a complex system is already not trivial. Recommending an agile investment project strategy to deliver that complex system adds a layer of analysis difficulty. To address the challenge, we need a well-defined system architecture and quantitative decision analysis.

This thesis demonstrates how to evaluate an agile investment project strategy, communicate its advantages and disadvantages and explore design variations. As a valuation example it compares an upstream oil and gas project designed with and without agile investment capability. The analysis utilizes known system architecture modeling languages to frame the object-oriented program. When executed the program simulates many successive annual decision scenarios. The numerical outputs and investment decisions can be easily explored. The program developed is flexible, it can be expanded and refined for other use cases.

The next two chapters introduce existing research and tools used for real options analysis and system architecture modeling. In chapter 4, we combine the introduced concepts, applying them to an upstream oil and gas system. We communicate the system architecture using modeling languages. We then create an object-oriented program, using Python, to simulate implementation and operations. Chapter 5 and 6 explore a hypothetical, but representative case. We use the program to compare architectures to develop an onshore, small oil field, including those enabling an agile investment strategy. The last chapters discuss limitations, program adaptation opportunities and conclusions.

Chapter 2: Real Options – Background

Real options and analysis methods

Real options was first introduced by Stewart Myers in 1977 in the context of company valuations as "opportunities to purchase real assets on possibly favorable terms" (Myers, 1977). They differ from the more common concept of financial options. A financial option's underlying asset is financial, objects that are regularly bought and sold: bonds, commodities, gold, etc. A real option's underlying asset is physical but concern non-financial decisions, for instance expanding a parking lot, shutting down a factory, or developing a new technology.

Traditional capital budgeting models are based upon a discount cash flow (DCF) analysis. Typically, metrics like positive net present value (NPV), internal rate of return (IRR), profitability index (PI) or payback period are used or combined to support decision making.

At a high-level, discount cash flow analysis considers capital invested, operational expenses, revenue and a discount rate. In the context of a project the objective is to evaluate the total life cycle cost of the asset, discounted under the time value of money assumption. Monetary value reduces as time progresses. The discount rate used for internal project comparisons typically differs from rates used to value entire firms. For projects within a company there is benefit in using a higher rate, since the cash that is invested in the project could have been invested in a more profitable project. Typical discount rates for project comparison range from 10-20%. Companies may have different strategies or standards for defining the rate used.

There are two main weaknesses of the traditional discount cash flow method. A core assumption is that all model inputs are certain. Forecasts for demand, project or operating costs, and even discount rate are assumed deterministic. There may be sound reasoning and complex modeling to extrapolate historical data or forecast future patterns, but the input for a discrete time increment is assumed to be a fixed number. Time series data for a complex physical system is challenging to predict with these modeling tools. Adding on the uncertainties of societal complexity to a system, accurately predicting time series forecasts of sociotechnical systems is extremely challenging.

To overcome the uncertainty in forecasts, discount cash flow analysis may rely on the average of each of the inputs to provide the average of the outcome value. Assuming this is always true is flawed thinking and has been coined as the "Flaw of averages" (Savage, 2009). Summarizing the idea, uncertain inputs can be thought of as a distribution of potential values.

When used in the model, the outcome may behave differently if the input value is not the distribution's average and is a low extreme or high extreme.

For example, a model's input is demand and the outcome is production. A manager may model that demand will equal production on average. However, if the assembly line can only produce 100 parts, but demand is 150 (not often but with some probability), the line will never deliver the extra 50 parts. Thus, average production will never be the average of the demand input, since there is no ability to take advantage of the upside of demand.

The second weakness in the traditional discount cash flow valuation is an all or nothing assumption which takes decision making power away from the future manager. Typical models will include an upfront capital cost, that could be spread over the first few years of a project's life cycle. Or if evaluating a phased investment plan, the years with capital expenditure are explicitly decided. What this assumption does is implicitly state that the entire project must be invested in during those early or determined years. It does not consider other future factors such as change in demand, cost or circumstance and then the management's ability to consider these factors. Future investments to either expand or abandon an asset can be included but the scenario is only evaluated under one version of future conditions.

Real options logic and evaluation techniques work to address these weaknesses seen in traditional capital budgeting discount cash flow methods. There are four main methodologies for option valuation that have been developed and used in practice (Ghahremani et al., 2012).

- Black-Scholes option pricing model
- Binomial Lattice models
- Decision tree
- Monte Carlo simulation

For financial options, the Black-Scholes pricing model can be used to evaluate European options where the exercise date is predetermined. There are assumptions used in this model that are weaknesses for applying it to real options modeling. The Black-Scholes formula is used for European options, or options that can only be exercised at a certain date, not prior. In a real option the expiration date may be known, but more likely it could be exercised earlier or later than this date. The exercise date is uncertain. Additionally, the strike or exercise price and the price of the underlying asset are also better defined in financial options using the market. The risk-free interest rate is also needed, which is typically approximated by short-term government bonds. The last large assumption it uses is a calculation for volatility, it uses an implied volatility

which can be derived using the market's forecast. In real options with a publicly traded market, there is not as clear of a proxy for implied volatility.

Another common financial options analysis is the binomial lattice pricing model. This assigns probabilities, based on historical observation, to the underlying asset price of increasing or decreasing to a discrete price at a discrete time. Then using these probabilities and the risk-free interest rate an option price is calculated. An advantage of binomial lattice is that input changes over time, such as volatility, can be incorporated into the model. There are still disadvantages when applying to real options, since inputs similar to Black-Scholes are needed for each discrete time interval as.

The key assumption for the Black-Scholes and binomial lattice models is that management decisions do not impact the "state of world", making them inapplicable for real option valuation. Other than exercising the financial option, there is no other actions that can be taken to combat market uncertainty. Whereas a real option does precisely that, a decision maker can choose to change the system's performance. The business manager can react to changing conditions by changing the system.

Decision trees can be thought of as an expansion of a binomial lattice and permits the inclusion of system changes. Using decision nodes, this method allows management decisions to be explicitly incorporated into the model. Uncertainty nodes can be modeled as low, medium, or high probability branches (sometimes referred to as P10, P50, P90) or with any other discrete probability combination. Decision trees have the advantage of interpretability and are essential in practicing decision analysis. The main disadvantage is that in order to incorporate several uncertainty inputs and decisions nodes, the tree grows exponentially since the model must be expanded to include every outcome. A very large tree reduces interpretability and challenges the ability to compute all branches. The computation of the full tree can be cumbersome even if using a digital technology.

Monte Carlo simulation is a fourth method that can be used to evaluate real options. The uncertainty inputs can be modeled as either discrete probabilities or probability distribution functions. Decisions are encoded as rules into the model. The model simulates many trials, discretizing the uncertainty distributions by taking randomized values from the distributions and applying the rule-based decision logic, to evaluate many outcomes. From this large set of outcomes their probabilities can be calculated as a representative distribution. The advantages are that many uncertainties and decisions can be simulated.

One of the disadvantages of Monte Carlo simulation has been the computing effort of running trials, however with the exponential increasing in computing power, software capabilities and programming packages this is of decreasing concern. Another disadvantage of Monte Carlo simulation is the interpretability for the decisions that are made, since they are formulated as rules, vs. explicit decision nodes as in decisions trees, understanding of the decisions made in each simulation run can be challenging to communicate.

Real options in a project

A summary model for real options decision making, Figure 2, and categorization of uncertainties, Table 1, and conditioning factors, Table 2, were developed in "25 Years of Real Option Empirical Research in Management" (Ipsmiller et al., 2019). Conditioning factors are used to improve option decision-making within the model.

All options (financial or real) can be thought of as having the right, but not the obligation, to make a future decision. Although the real options concept was first introduced by Myers as an asset to purchase, the concept of real options can be expanded to the design of a physical asset. The original definition can be thought of an option "on" a physical asset, the right to invest capital into a project or the right to abandon the project. Based on the Ipsmillar et. al. research this type of real option is the most well-known and researched.

Expanding the concept into design flexibility, the concept can be thought of as a real option "in" a project. This type of flexibility is built into the physical design of the project or system (de Neufville & Scholtes, 2011). The framework and examples in Table 1 and Table 2 are based on the real options "on" a project. This thesis will describe and use an adaption of this model for design flexibility as a real option "in" a project. Figure 2 is an adaption of a real option decision making research model (Ipsmiller et al., 2019) for options within a project developing a physical system. It includes flexibility within the design as a moderator and the options structure is analogous to the system's architecture (Chapter 3).

Examples of design flexibility could include expansion capability, turn-down capability or enabling future technologies or systems to be integrated into the system over time. These examples have been added to Table 2. These aspects of design typically come at an increase in the initial capital cost of the project. Since this cost is incorporated early over the project life cycle, traditional DCF methods will devalue the worth of the overall project since capital cost has increased but has no mechanism for added value based on future decisions. This thesis will explore evaluating a complex system development with a strategy of agile investment. The

capital can be invested on annual basis to expand the system. The investment decision "rules" being coded as an optimization algorithm and the uncertainty in inputs simulated using Monte Carlo methods.

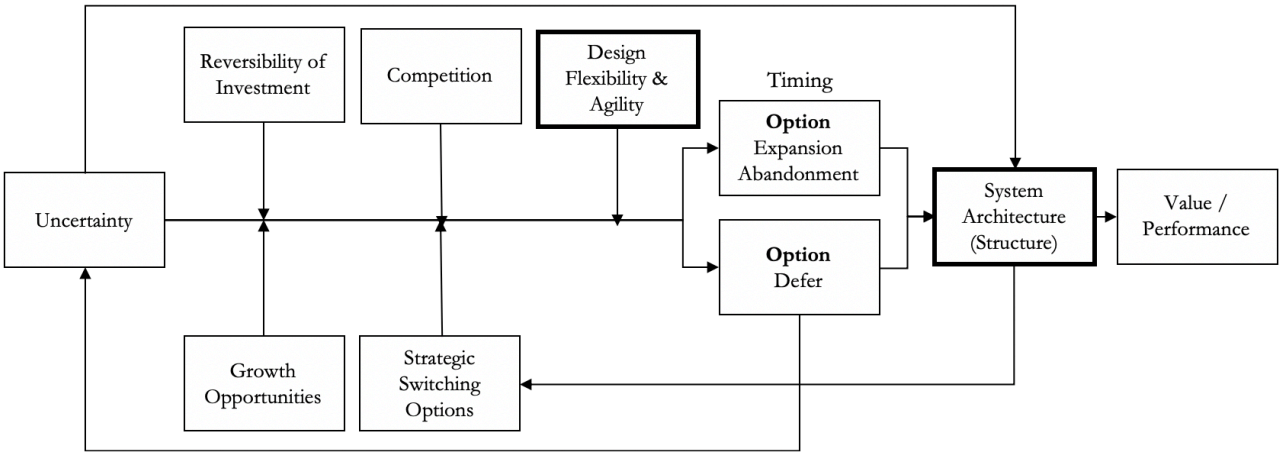

Figure 2: Adapted model of real option decision making for real options in a project

Types of Uncertainty			
Demand uncertainty	**Technology uncertainty**	**Macroeconomic uncertainty**	**Partner uncertainty**
Examples			
Industry uncertainty	*Industry R&D intensity*	*Price volatility*	*Political risk*
Stock exchange volatility	*Perceived technology uncertainty*	*Country risk*	*Perceived threat of opportunism*
Industry production volatility	*Technical experience*	*Exchange risk*	*Prior experience with partner*
Industry demand volatility	*Technical knowledge*	*Institutional/Cultural*	*Public/Private firm Sample*
Perceived demand uncertainty	*Technical distance*		
Target industry experience	*Self–employment experience*		
	Entrepreneurial experience		

Table 1: Types of Uncertainty (Ipsmiller et al., 2019)

Panel of Conditioning Factors				
Irreversibility	**Competition**	**Growth Opportunities** (Macroeconomic)	**Strategic Flexibility**	**<u>Design Flexibility "IN" project</u>**
Examples Types				
Size of investment	*Size of Competitors*	*Growth potential*	*Foreign experience*	Expansion capability
Perceived irreversibility	*Competitor Commitment*	*Competitive advantage*	*Product experience*	Turn down capability
	First mover		*Exchange rate*	Location mobility
			Portfolio	Facilitation of other technology
			Ownership	

Table 2: Real option conditioning factors (Ipsmiller et al., 2019). Design Flexibility "IN" was added as a moderator.

Chapter 3: System Architecture – Background

The systems engineering "V"

The International Council of Systems Engineering (INCOSE) defines systems engineering as a "transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods." (International Council of Systems Engineering (INCOSE), n.d.). Systems engineering follows the system "V" model, one adaptation is shown in Figure 3. Diagramming the sequence of major stages in a system's design, development and implementation. In this adaption system lifecycle processes are also included.

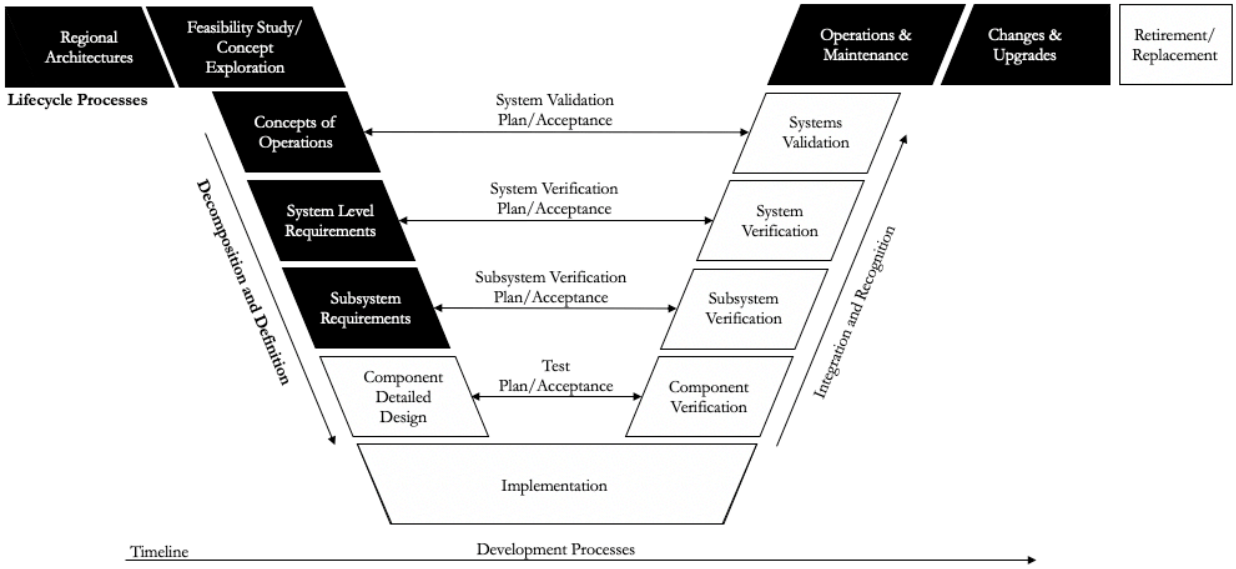

Figure 3: Systems Engineering "V" Model. Thesis focus areas in black.

The system "V" model aligns well with a traditional discount cash flow analysis, discussed in Chapter 2. The upfront capital investment is used to complete one iteration of the "V". When the system moves into operation and maintenance revenue and operating expenses incur.

A system designed for the strategy of investment agility, could be thought of expanding the "Changes & Upgrades" stage. This stage could be decomposed into additional "V's", executed in sequence if the real option is executed (Figure 4). The key difference is incorporating these additional "V"'s into the system's concept evaluation and other earlier stages of project design development. This incorporation could have a significant effect on the recommended system architecture and overall capital investment strategy.

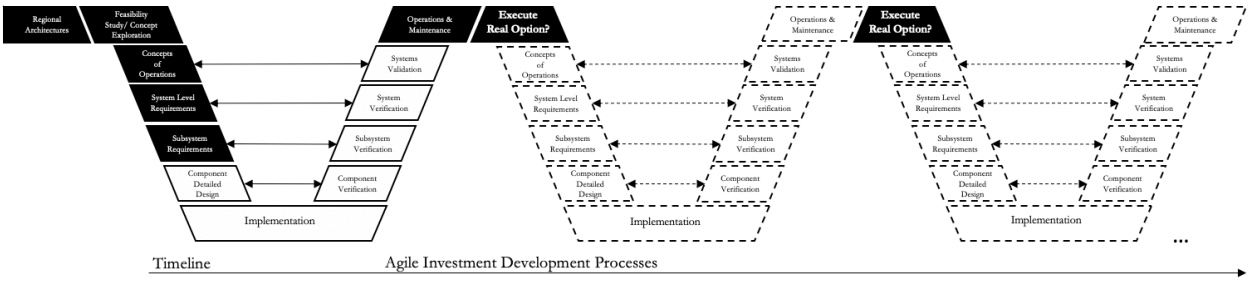

Figure 4: Systems Engineering "V" Model for Agile Investment. Thesis focus areas in black.

System architecture modeling

In systems engineering the value and benefits derived from a complex system are defined early on during the development of the system architecture. This definition of value drives the system's subsystem and component forms to create their functions and interaction. Numerous studies prove that the early definition and design stages of a project are the most influential in driving overall success of the system (Tan et al., 2017).

System Architecture can be thought of as the system's framework. The architecture encompasses the early stages of the left-hand side of the "V" model: understanding the context and regional architectures; concept exploration and operations; and definition of system and subsystem level requirements (highlighted in black in Figure 3 and Figure 4). Since system architecture strives to build the basis for later design development and implementation steps, there has been a focus on standardizing tools for communicating system architecture knowledge. These tools include modeling languages and diagrams for visually explaining architecture concepts to stakeholders and future teams. This is important since future teams, especially for larger systems, may not have been involved in the creation of the system architecture. These teams work on downstream stages of detailed design, implementation and operation.

A key concept in communicating system architecture is the separation of function and form (Crawley et al., 2016). Object Process Methodology (OPM) is a modeling language used to create models that communicate context, knowledge, and design of complex systems. In this graph language, nodes are either the concept of an objects (form) or the concept of their behavior (process or function). Edges are used to connect the nodes signifying different types of relationships between concepts.

Developing an Object Process Diagram (OPD) using OPM that clearly communicates the value, form, function and interactions can be challenging for complex systems. The OPD creator must employ a level of abstraction. Too much detail and noting every single object and its

behavior can distract from clearly describing the system's framework. It is typical to develop an iterative approach in finding the "right" level of abstraction for communication.

Decomposition is one of the methods employed for this. The system is first decomposed into subsystems, the subsystems are further decomposed into components. This is known as "in-zooming" in systems engineering. Using additional modeling tools and analysis like decomposition diagrams, component interface diagrams and design structure matrices (DSM), insights may be discovered for subsystem grouping of components. For example, highly coupled component interactions currently in separate subsystems or component functionality that is similar in another component.

Mentally moving out of the decomposition from the component level to the subsystem level ("out-zooming") with these new insights may lead to rearrangement of the initial groupings. Analysis of subsystem decomposition can lead to clearer definition of subsystem requirements, creative component selection to meet those requirements, easier more organic detailed design development and reduction in interface management (Crawley et al., 2016).

Systems Modeling Language (SysML) is another language used for modeling any type of system, although it was adapted from a software engineering specific modeling language, Unified Modeling Language (UML). Unlike OPM, SysML has nine distinct types of diagrams. These diagrams cover structure, function and requirements. SysML's detail and allowance of additional diagrams makes it a good tool for further system architecture definition and discovery.

Model based system engineers have adopted both OPM and SysML. The advantages or disadvantages of the languages have been debated by practitioners and research groups. Arguments have been made that OPM can clearly communicate a holistic understanding of the system and its context quickly. They argue SysML diagrams are too complicated for quick insights. SysML advantages include the ability to represent the system in greater detail, different perspectives creates traceability and representation of more information (Grobshtein et al., 2007).

Both OPM and SysML languages can be leveraged to visualize, communicate and gain insights into a complex system's architecture. However, to enable computational analysis of the architecture another tool may be better suited. Object oriented programming (OOP) is a classification for programming languages based on creating object concepts. The objects have methods and attributes containing data. The objects can be specific to the domain that the program is written for. The behaviors or functions of the objects are referred to as methods, these methods can access an object's attributes' data and perform behaviors to transform the data. The

methods can also be functions that bring sets of attribute data together and perform an analysis to create or modify another attribute.

Class-based OOP uses the concept of classes to group objects. Class creation defines the attributes and methods for an object type which provide the structure for “instances” of the class. Instances are specific creations of the object, i.e. class: Person, instance of class Person: John Doe. Instances are created by defining the class attributes needed and they inherit the behaviors (methods) created in the class definition. i.e. class has the attribute first name and method walk, John Doe has the attribute first name John and has the ability to walk.

There are parallels in OOP, OPM and SysML with their three concepts of objects, processes and interactions of the objects. However, they were developed for different uses in representation and analysis. There are benefits for using all three in system architecture design and decision making. In this thesis, we utilize all three, leveraging their similarities to recommend a system architecture that can be communicated as well as evaluated.

Chapter 4: Case Study - Upstream Oil & Gas System Modeling

An upstream oil and gas system is complex and there are different strategies that can be used to develop it. In full context, this system is only a part of the larger energy system that delivers fossil fuels to consumers. The upstream system's primary benefit is the energy supplied to downstream systems in the form of petroleum fluids. Its primary value to the company operating the system is the profits derived from the sales. The fluids of crude oil, natural gas liquids and produced gas are transferred and or processed by systems in the midstream sector. A midstream system's primary function is to transport the separated fluids to systems in the downstream sector. In addition, midstream facilities further separate the produced gas into natural gas and natural gas liquids for transfer. Downstream systems perform additional fluid processing to create market products, such as jet fuel and gasoline.

System architecture concept model in OPM

The system architecture for an upstream asset can be described at a high level using the OPM language introduced in Chapter 3. Figure 5 reads from left to right, objects are in rectangle nodes and processes in oval nodes. The objects and processes are separated into swim lanes, with the lanes to the right moving away from the value products directly involved in producing the overall intended system emergence of crude oil and produced gas sales.

The edges indicate links or relationships between objects and process. Arrow links are considered transformational and indicate an object is being transformed or acting on by a process. An arrow extending from an object to a process can be read as the process "consumes" the object, i.e. Investing consumes Capital. In contrast an arrow from a process to an object indicates the process "yields" the object, i.e. "Extracting yields Reservoir Fluids.". A double-headed arrow can be read as the process "effects" the object, i.e. "Investing effects the Reservoir". The object connected to a process with a transformational link is called the object operand.

Objects can also be instruments, or the form that enables the process. This object-process link is called an instrument link and is indicated by an edge ending with a circle at the process. It can be read as "handles" or "requires". For example, Extracting requires Well & Production Equipment.

Structural relations are indicated with triangles and are a one-to-many relationship. A solid black triangle can be read as "consists of", i.e. Revenue consists of Produced Gas Sales and

Crude Oil Sales, but not wholly since Operating also effects Revenue. A white outlined triangle relationship indicates an attribute is “exhibited”, i.e. Gas Processing Equipment exhibit the attributes of Compressing and Pressure Releasing.

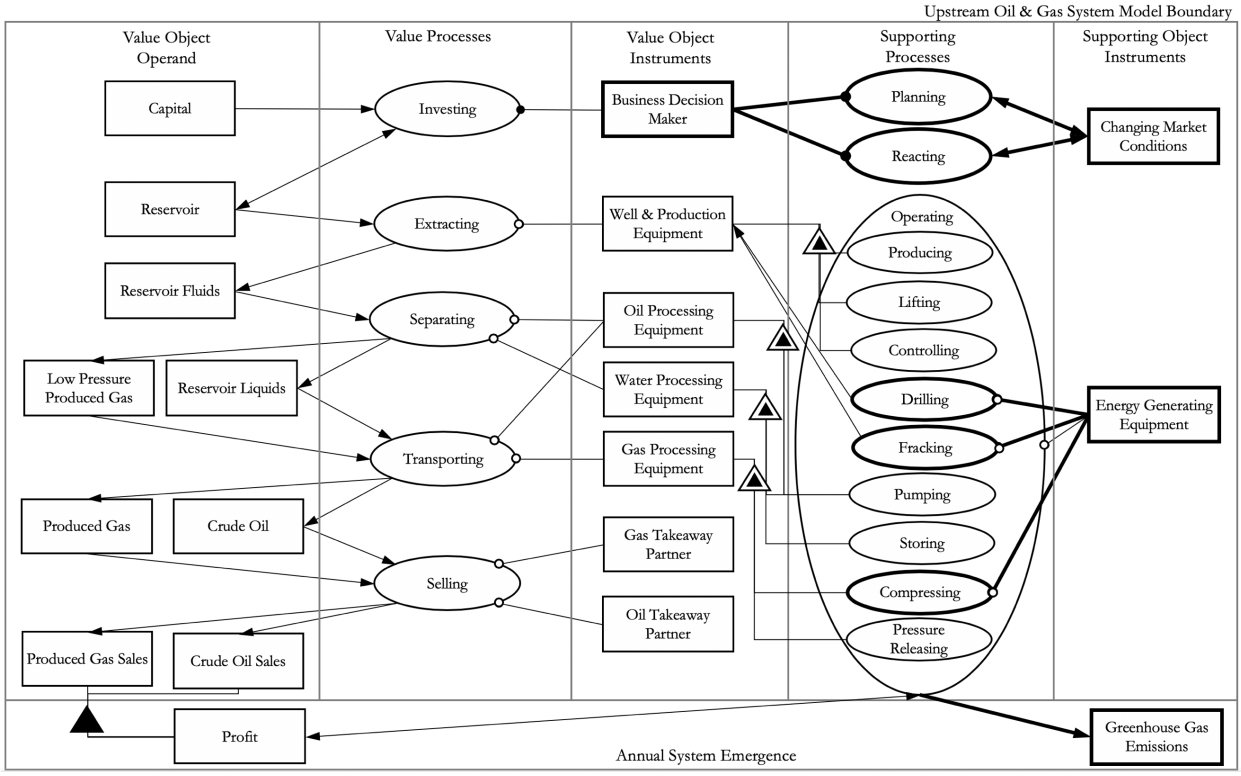

Figure 5: System Architecture Object Process Diagram

We can use OPM to highlight priority areas of the system and reduce its overall complexity. Investing in capital creates annual cashflow from oil and gas sales, by first extracting the reservoir fluids, separating the gas and liquids, transporting and selling the value products of produced gas and crude oil. The physical work on the fluid is enabled by the equipment used. This equipment is included in a conventional description of an upstream oil and gas system: oil, water, gas processing and the partners the value fluids are sold to. Operating the physical equipment also effect the overall revenue of the system.

In Figure 5 we describe a very typical upstream system architecture. These concepts are to be included in a decision analysis model calculating economic value for an investment decision. Many project valuations will use a one-time capital equipment cost in the first years of the project timeline. Subsequent revenue and operating expenses are calculated post investment.

Differentiating the system architecture for this case are two subsystems, objects and processes highlighted in bold. These become apparent as the system architecture is expanded to include supporting processes and instruments. These supporting subsystems depend on or are

additional sources of uncertainty. Incorporation of their potential uncertainty and interactions with the system may drive different outcomes in performance. The differences have the potential to affect the system architecture and the business decisions that are made.

The first of the additional concepts to be included is the changing market conditions. They effect the planning and reacting processes of the business decision maker. This is the primary subsystem that the real option decision will rely on. Unlike a one-time project valuation, a decision maker's ability to invest or not to invest is continuous and dynamic. This idea also emphasizes a disadvantage for communicating the system architecture through Figure 5 and the challenge in evaluating design flexibility. The capital, the operands and the instruments themselves will change dynamically over time during the valuation timeframe.

The second concept incorporated into this case are the undesired greenhouse gas emissions created from operating the physical system. As a supporting instrument, energy generation equipment may not be treated as an architectural decision in typical system design or at minimum, its emissions may have a secondary role in the overall capital investment decisions. As government policies and business strategies concerning environmental impact become increasingly uncertain, this supporting instrument may warrant increased attention in architectural analysis. Although sometimes incorporated as a carbon price into an overall economic valuation, a separate emissions metric could influence the system's architecture.

System architecture concept model with SysML

The concept model in Figure 5 is a good start for communicating the architecture of the system and paints a clear architectural story. However, the level of abstraction does not give enough detail to set up an object-oriented programming model to perform an analysis. We found further detailing of the system architecture is needed to communicate specifics and create the insights needed to develop the program's framework.

Starting with a SysML block diagram the system decomposition can be represented. We decompose the overall system architecture into eight functional subsystems: Well Production, Oil Processing, Oil Takeaway, Gas Processing, Gas Takeaway, Water Processing, Energy Generation and Investment Decisions (Figure 6). Each of these subsystems are further decomposed into Level 2 components. This diagram has more detail than Figure 5 displaying granularity in the components used as instruments for enabling the supporting systems. However, it does not describe system value and provides less information on concept relationships.

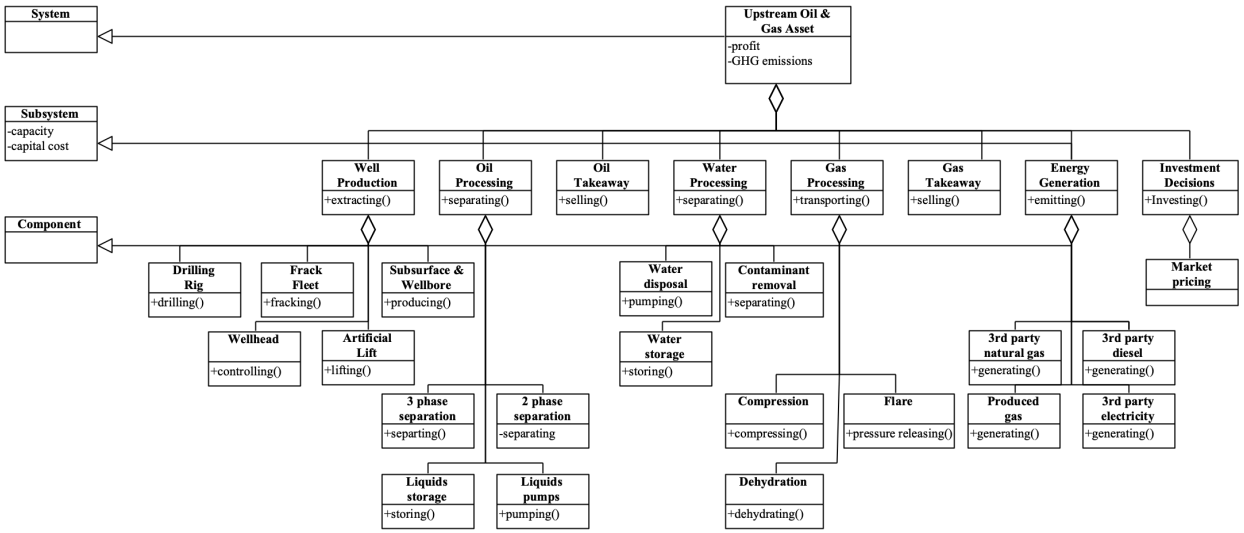

Figure 6: System Decomposition Model

The well production system is typically designed by geologists, petroleum engineering and drilling-completion engineering teams. The components and engineering details are based on the geological mechanics and location of the petroleum reservoir. For this case the wells' changing production will be used as input demand for the rest of the system design. The well production subsystem is decomposed into four components. The wellbore is drilled by the drill rig and the rock is fractured, assisting fluid flow, by a frack fleet. The wellhead and artificial lift are installed at the surface of the wellbore. The wellhead controls the pressurized production fluids from the wellbore. Artificial lift equipment is used to enhance production recovery.

The oil processing's main function is to store and separate the crude oil from the water and produced gas. It has been decomposed into 4 components, 3-phase separation is typically used for initial separation of oil, water, gas. 2-phase separation is used to further separate oil and water which are stored in tanks, prior to being pumped to the oil takeaway subsystem. Oil takeaway in this case is the crude oil interface to the midstream system.

The water processing's function is to dispose of the water. This involves three components, further contaminant removal, storage and then water disposal. Water takeaway is typically a system to reinject the produced water into water disposal wells.

Gas processing typically involves compression and flaring of the produced gas. It can be decomposed into 3 parts. Dehydration removes water from the gas so it will meet specifications for comingling of the gas from other sources in the takeaway system. Compression enables the produced gas to enter the gas takeaway's pipeline system. Flaring is necessary for safe

operations. In case of planned or unplanned system upsets, the flaring system is used to safely depressurize the system by burning and releasing the produced gas to atmosphere. Gas takeaway is the produced gas interface to the midstream partner's gas plant. The gas plant will further process the produced gas in order to separate it into natural gas and natural gas liquids.

Expanding the model

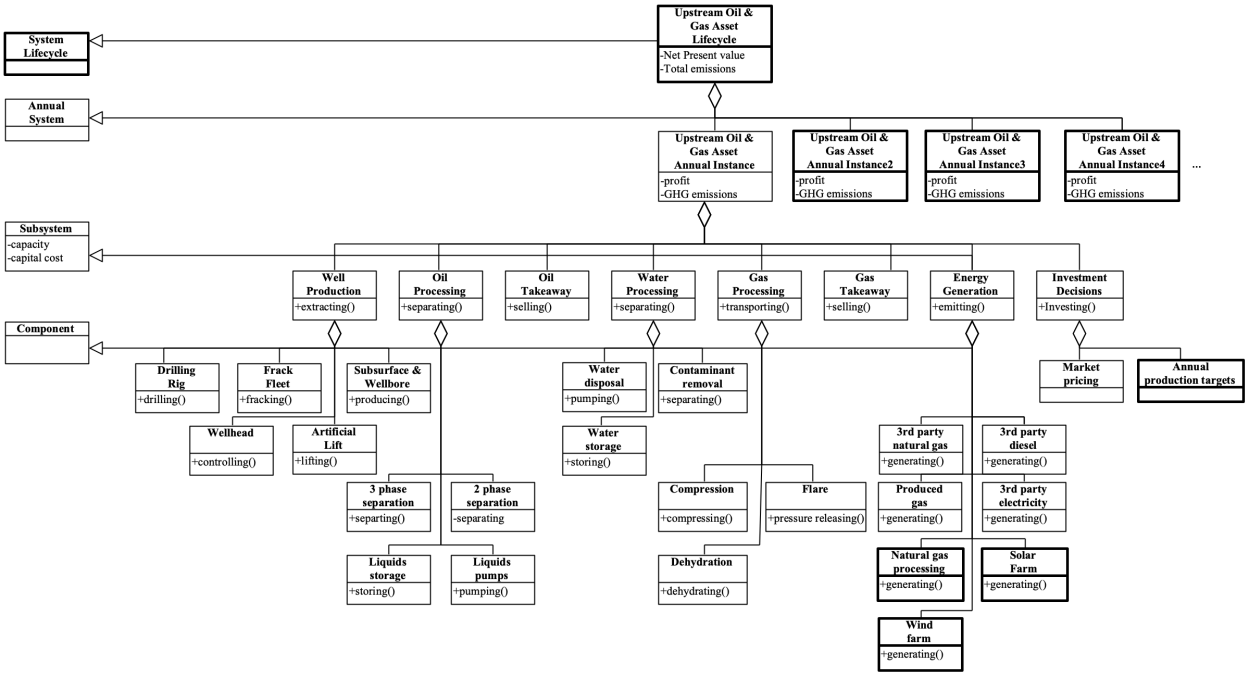

Figure 7: System Decomposition Model Expanded

Adding additional component options to reduce combustion emissions

The well production, oil, water and gas processing subsystems make sense to focus on during the shaping and design phases of a project since they directly impact the value operands, (Figure 5). As energy generation components enable supporting processes and are further from the value products, these would not be in the forefront of architectural decision making. However, we are using greenhouse gas emissions as a key performance metric for the system so are including the subsystem in the expanded decomposition (Figure 7).

Diesel and gas engines and generators emit a large portion of the total greenhouse gas emissions created in an upstream development. In 2009 the United States Environmental Protection Agency implemented the mandatory reporting of GHG emissions for large sources, over 25,000 metric tons of carbon dioxide equivalent (CO_2e) per year (United States Environmental Protection Agency, 2009). The Greenhouse Gas Reporting Program (GHGRP),

under subpart W, requires emissions from portable and stationary combustion equipment used in onshore production.

The case will model three of the largest and most consistent energy consumption components in an onshore, unconventional asset that use combustion: drilling rig, frack fleet and compression. A drilling rig demands large amounts of energy, typically supplied by diesel generators. A frack fleet consists of many pieces of portable equipment, used to hydraulically fracture the shale rock to increase fluid flow. The high-pressure pumps, used to inject liquid to open rock fissures, demand large amounts of energy. Frack fleets have also historically used diesel engines to generate the energy needed. Recently both drilling rigs and frack fleets have seen an increase in using electricity generated by natural gas to supply energy, mostly due to changing market conditions with low natural gas prices.

Compression is needed to transport produced gas to the gas takeaway partners' system. Upstream field compression subsystems use gas engines directly or gas generators for electricity to power the reciprocating or centrifugal compressor. Buy-back natural gas agreements with the gas takeaway partners or the field's produced gas are commonly used.

Diesel, natural gas and other fuel sources have differing emission factors used to calculate CO_2e. (United States Environmental Protection Agency, 2018). Pipeline quality natural gas, or natural gas downstream of a gas plant is considered a cleaner burning fuel than diesel. Produced gas still has heavier hydrocarbons entrained in the gas and thus emits more emissions when burned.

The gas processing technologies of natural gas liquid extraction are used in the takeaway subsystem, usually as components of large scale processing facilities operated by the sales partner. At a midstream gas processing plant, the gas is further dehydrated (water removed) and separated into gas products. Natural or residue gas is what most consumers are familiar with and is mostly made up of the lightest of the hydrocarbon chains, methane. The heavier hydrocarbon chains (i.e. butane, propane, ethane, etc.) are extracted from the produced gas as a mixture of natural gas liquids (NGLs). This is typically done by cryogenic turbo expander technology or absorption technology using an absorption oil (Hubbard, 2009). Using fractionation, the mixture of NGL's is split into viable products for market.

Other midstream facilities downstream of a gas plant may employ compressed and liquified natural gas processing. These technologies reduce the natural gas volume creating cost effective alternatives for shipping and transport to new markets. Compressed natural gas (CNG)

is stored at ambient temperature at a high pressure, greater than 3000 psi. Liquified natural gas (LNG) equipment cryogenically cools the natural gas into a liquid at -260 °F. Both fuels have a current target market, CNG is being used in land transportation vehicles while LNG is used to ship large quantities of natural gas to geographically far markets.

The potential to employ these technologies in an upstream environment may provide a cleaner fuel source. For fields that do not have close access to buy-back natural gas from a gas plant, this reduces additional pipeline infrastructure. Processing produced gas closer to the combustion source also frees up capacity in systems that may be constrained by uncontrollable partner takeaway capacity. However, the processing would be done to meet internal energy demand so the additional gas processing equipment must be scaled down. A typical midstream gas plant has a capacity on the order of hundreds of millions of cubic feet per day, taking in produced gas from many upstream partners. Recently small scale or micro-scale gas processing technologies have reached commercial markets and implemented in limited use (Global Gas Flaring Reduction Partnership, 2019). We will look at the potential use in portable combustion equipment used in drilling rigs and frack fleets. This makes small scale CNG technology downstream of NGL extraction a potential lower emission option for energy generation.

Solar and wind renewable energy technologies are not commonly used in conjunction with traditional oil and gas development. They are considered an alternative to the value product that the upstream system produces. However, there are energy demands and consumer products that cannot directly use electricity produced from renewables and a hydrocarbon fuel is needed. Although seemingly counterintuitive, renewables can be used to reduce the emissions created from hydrocarbon production. This can reduce the overall emissions of the entire hydrocarbon value chain, from production to direct consumer use.

The decomposition shown in Figure 6 can be expanded to include additional components to lower emissions during energy generation. These are not conventional in an upstream oil and gas system. This includes energy generated from cleaner natural gas processed from the system's own produced gas, solar farms and wind farms. These additional component types are highlighted in bold in Figure 7.

Accounting for expansion decisions over time

We've expanded the SysML block diagram to also introduce investment agility via expansion decisions over time (Figure 7). Instead of evaluating one-time investment decisions based on a long-term projection of market conditions, investing decisions can be simulated

annually based on changing conditions. These decisions add complexity to the system architecture concept and evaluation. The system's attributes and subsystems may change dynamically, and these changes must be incorporated into the next decision.

Not only do the invested subsystems change over time, but the amount of oil, water and gas produced from the well production subsystems also change. Commonly referred to as a decline rate, lower production in existing wells open up capacity space in the processing and energy generation subsystems, therefore also effecting the annual investment decisions.

Additional complexity also comes from the difference in timescale for subsystem expansion. The time frame from decision to operational readiness for the subsystems varies greatly. Drilling and fracking onshore is a matter of days or weeks, while executing and installing other subsystems can take many months or many years depending on its size and type.

Historically large oil and gas facilities that incorporate all of these subsystems would take more than a year to design, build, execute and ready for operation. A system development like this could be thought of as one cycle through the systems engineering "V". The system architecture implementation changes drastically when looking towards a system developed over time with investment agility. By standardizing, subsystems design and procurement planning steps can be reused and optimized, reducing decision to operation turnaround time.

For this case we will incorporate the use of standard size oil and gas processing subsystems. The decision is not only to invest in expansion of the subsystem but also which standard size to use. Setting up the program with decisions based on size lets us refine and explore subsystem capacities. By varying the subsystem sizes, costs and schedule we can continue to evaluate the impact of these adjustments on the overall system performance.

Using the timescale for operation readiness, we decomposed annual expansion investment decisions into two main decisions. Well Production expansion and other facility expansions. For this case investment decisions that expand the capacity for gas processing, oil processing and energy generation are realized one or more years after the decision is made. Therefore, these decisions are made based on the next year's production target. Well production investment is realized in the same year striving for the current year's production target but is constrained by existing system's processing capacities. The annual production targets are added in bold in Figure 7, to show they also are a component of investment decisions.

Setting up the decision-making timeline on an annual basis allows the expanded system's lifecycle to be evaluated. Evaluation over the life of the system can be done by evaluating the

annual “snapshots” of the system. Figure 7 shows these snapshots highlighted in bold as “Upstream Oil & Gas Asset Annual Instance 2-4…”. Each of these instances can be decomposed into its own subsystems and components like the first annual system instance.

A system lifecycle is made up of these annual instances. Each annual instance has its own annual cashflow and emissions. These values can be used to evaluate the overall system lifecycle performance for net present value and total lifecycle emissions. Additionally, each annual instance can be evaluated for other metrics like specific subsystem spend, expansion decisions, energy required, etc.

The system architecture diagrams help communicate the concepts needed to simulate performance of the architecture. By developing these models, we gained insights for structuring the object-oriented program. To incorporate real options the model needed to be expanded to show changes in the system. Additionally, each subsystem has specific calculations or values for determining overall system performance. These values change with the components that make up the subsystem. Therefore, the program must be able to account for different types or sizes of subsystems, even if the subsystems have similar behavior. The final insight gained is that multiple decisions are needed based on installation to operation timing. Consequently, both shorter term and longer term production targets are needed.

System evaluation model in Python

We used the object-oriented programming language Python to translate the system architecture into a simulation to explore system performance. The program used the objects created during architecture modeling to represent the domain classes of the system. The inputs of the program include necessary attributes of the class objects, including deterministic attributes and attributes with uncertainty. The output of the program is a system lifecycle instance, with attributes of net present value and total emissions, as well as the annual system instances that were simulated during computation.

Evaluating a multi-stage expansion project is challenging since there are consecutive decisions to be made over the life of the project and these decisions are based on future uncertainties. For this case the decisions have been simplified to three annual decisions: 1. Oil pricing effect on production targets 2. Expansion of Well Production systems and 3. Expansion of the other primary value subsystems (oil processing and gas processing) being grouped as facilities. There are eight Python modules that comprise the program:

- `main.py` : Defines the inputs to start the simulation and executes the deterministic and Monte Carlo simulation analysis. Calls functions embedded in the other modules.
- `systemClasses.py` : Defines the object classes used to describe the system domain. Parent and child classes are defined to enable instances of the objects with attributes and access to methods that the object can execute.
- `priceForecast.py` : Defines the expected annual price forecasts and probability distribution functions used to simulate crude oil, natural gas, natural gas liquids and diesel under uncertainty.
- `annualSystemDecisions.py` : Contains the function to create an annual system instance object. Simulates annual decision making and production by calling class methods, price forecast functions, wellsToDrill.py and facilitiesToInstall.py.
    - `wellsToDrill.py` : Contains the function to simulate an annual decision determining the number of wells production subsystems to drill and fracture. Uses an integer optimization program solver. Objective is to maximize well production constrained by the existing system capacity.
    - `facilitiesToInstall.py` : Contains the function to simulate an annual decision determining the number and type of oil and gas processing subsystem to install. Uses an integer optimization program solver. Objective is to meet next year's production target while minimizing the capital spent on the new facility capacity.
- `systemEvaluate.py` : Contains a function that calls methods of the System class to evaluate the system after annual subsystem expansion decisions are made.
- `visualizations.py` : Defines functions that create graphs to explore the object attributes and system performance after the simulation is run.

Appendix A provides further details for each of the Python files (modules). Excerpts of code are provided to support understanding of the program. Additionally, the two integer optimization algorithms are defined.

Chapter 5: Case Study - Basic Analysis of Agile vs. Deterministic

Structure of Case Study

We will use the Python program, described in Chapter 4, to evaluate development strategies for a hypothetical small onshore unconventional oilfield. Being onshore, the subsystem equipment is installed on the ground. Being unconventional, there is a need for fracking or fracturing the reservoir rock. Subsystems of well production, oil and gas processing, oil and gas sales and water processing (Figure 7) are all needed to develop the field.

The analysis explores the system performance when differing the amount, size and installation timing of the various subsystems. The field has a production target of 40 MBPD to be achieved by 2025 and maintained thereafter. Production targets are based on a deterministic oil price forecast, which will change as prices change in the uncertainty simulations. The system itself changes with the expansion of the system's infrastructure and as wells' production decline. Three initial alternatives for field development are compared:

1. Planned Capacity Architecture:

   Install one system with a nameplate capacity equal to that of the maximum deterministic oil production target. 1 'Extra Large' Oil Processing subsystem and 1 'Large' Gas Processing subsystem.

2. Large Scale Agility Architecture:

   Install trains with capacity half of maximum production target, 1 'Large' Oil Processing subsystem and 1 'Medium' Gas Processing subsystem. Revisit annually decision to install additional subsystems of same size based on production target uncertainty.

3. Multi Choice Agility Architecture:

   Revisit annually facility installation and size decisions. Choice of four oil processing and three gas processing types. Targeting minimum objective of meeting next year's production target while minimizing capital spend.

For each scenario drilling decisions (simulated as installation of a wellProduction subsystem) are revisited with the objective of maximizing the capacity of the existing system. Expansion for oil and gas processing is also revisited for scenarios 2 and 3, as described. The existing takeaway and water processing subsystems capacities are large enough to be considered unconstrained.

Input of Case Study

For all scenarios, the 2020 system starts with the same existing subsystems and potential future locations (Table 3). We start with the six horizontal exploration wells with current production of 800 BPD (to decline annually). A small tank battery performs the three phase separation for oil processing is also installed. There is no permanent gas processing installed, the existing equipment will be removed since it was meant only for temporary use during exploration. A water disposal well and partnerships for takeaway are existing.

2020 Existing Subsystems & Future Subsystem Placeholders							
Location Index	Subsystem Child Class Name	Type Description	Capacity	Units	Capital ($MM)	Year Installed	Other
0	wellProduction	horizontal	0.8	MBPD	5	2020	0.8 (current production)
1	wellProduction	horizontal	0.8	MBPD	5	2020	0.8 (current production)
2	wellProduction	horizontal	0.8	MBPD	5	2020	0.8 (current production)
3	wellProduction	horizontal	0.8	MBPD	5	2020	0.8 (current production)
4	wellProduction	horizontal	0.8	MBPD	5	2020	0.8 (current production)
5	wellProduction	horizontal	0.8	MBPD	5	2020	0.8 (current production)
6	oilProcessing	small tank battery	5	MBPD	8	2020	
7	gasProcessing	to be decommissioned	0	MMSCFD	0	2020	
8	waterProcessing	disposal well	100	MBPD	15	2020	
9	gasTakeaway	Partner A	200	MMSCFD	0	2020	0.3 ($/MCF processing fee)
10	oilTakeaway	Partner B	100	MBPD	0	2020	
11-30	oilProcessing	future	0	MBPD	0	future	
31-50	gasProcessing	future	0	MBPD	0	future	
51-350	wellProduction	future	0.8	MBPD	5	future	0 (current production)

Table 3: Existing 2020 System – initializer for scenario simulations

The system will be analyzed over a 10-year period, from 2021 to 2030. The deterministic production targets are in Table 4. The attributes of the oil and gas processing types are described in Table 5. Depending on the scenario, the annual choice for these subsystem's is limited or expanded to include the various types.

Assumptions were made to incorporate the project "iron triangle". The three constraints of scope, budget and schedule can be traded off, but changing one causes a change in at least one of the other two. For example, increasing scope, in this case capacity, from a medium to large tank battery will cause an increase in cost and schedule. Additionally, the attributes incorporate economies of scale, where scaling capacity has a cost advantage. For example, doubling the processing capacity of the medium tank battery, the large tank battery is only 1.6 times the cost.

Year	Target Oil Production (MBPD)
2021	5
2022	10
2023	20
2024	35
2025	40
2026	40
2027	40
2028	40
2029	40
2030	40

Table 4: Deterministic oil production targets.

Oil Processing Types				
Description	Tank battery Size			
	Small	Medium	Large	Extra large
Capacity (MBPD)	5	10	20	40
Cost ($MM)	8	14	23	40
Years to Operational	1	1	2	3
Gas Processing Types				
Description	Compressor Station			
	Small	Medium	Large	
Capacity (MMSCFD)	20	40	80	
Cost ($MM)	5	9	15	
Years to Operational	1	2	3	

Table 5: Oil and Gas Processing standard subsystem types to be annually chosen during scenario simulations.

Inputs with uncertainty

We chose inputs that the owner of the system has less control over to model with uncertainty. The market is a global system that controls commodity prices and is highly uncertain. Reservoirs and drilling can also be difficult to predict and their characteristics effect directly the production sales and emissions. Prediction is so complex specialized teams are dedicated to creating detailed, simulation models just focused on reservoir mechanics and well design. Uncertainty in uptime for the gasProcessing subsystem was also included. The number of days compression is available varies, changing the energy requirements and emissions of the system.

The case is hypothetical, but representative of current market data. Commodity prices are forecasted by a variety of resources and companies may have their own internal price decks to use. For our study we will use data from the World Bank Commodities Price Forecast, updated in April 2020. Uncertainty in oil, natural gas, NGL composite and diesel prices are modelled as annual forecasts with a gamma distribution (equations 1-3). Where the mean (μ) is the deterministic price (*World Bank Commodities Price Forecast*, 2020) and standard deviation (s) is a fraction of the mean. The gamma distribution is appropriate since annual average prices are assumed to never be negative and can be skewed to have longer positive tails. The program can easily be configured to use any probability distribution by changing the distribution function located in the priceForecast.py module.

Figure 8 through Figure 15 show the price uncertainty over the life of the system and a representative 2021 price distribution. The 2021 graphs plot the probability density output of calling the 'getPrice' functions for 1e6 simulations.

$$f(x) = \begin{cases} \frac{\left(x/\beta\right)^{\alpha-1} * e^{-\frac{x}{\beta}}}{\beta\Gamma(\alpha)} & x \geq 0 \\ 0 & otherwise \end{cases} \quad (1)$$

$$\alpha = \left(\frac{\mu}{s}\right)^2 \quad (2)$$

$$\beta = \frac{s^2}{\mu} \quad (3)$$

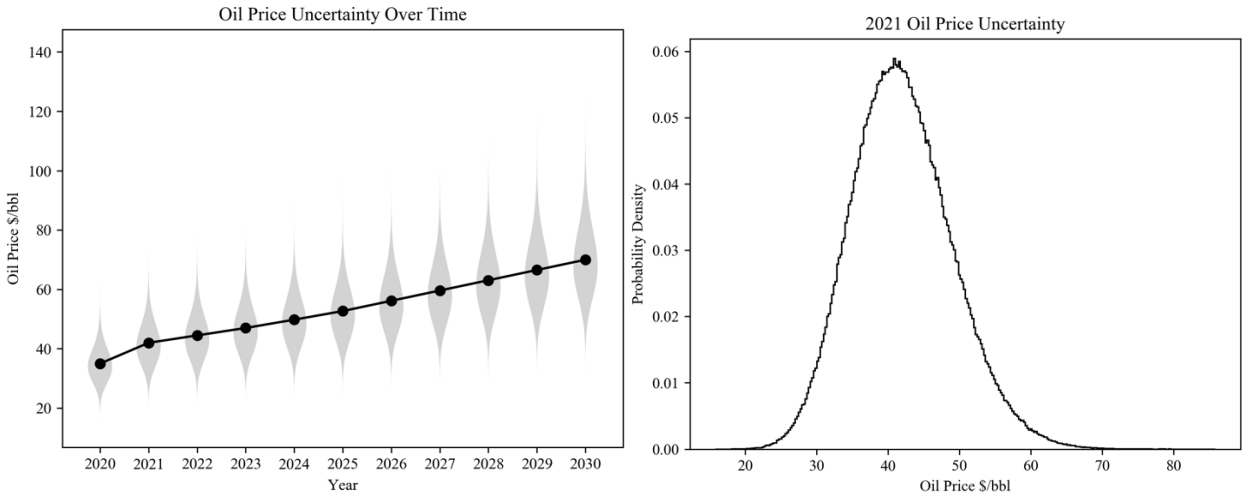

Figure 8 & Figure 9: Oil price uncertainty over time & 2021 probability distribution function

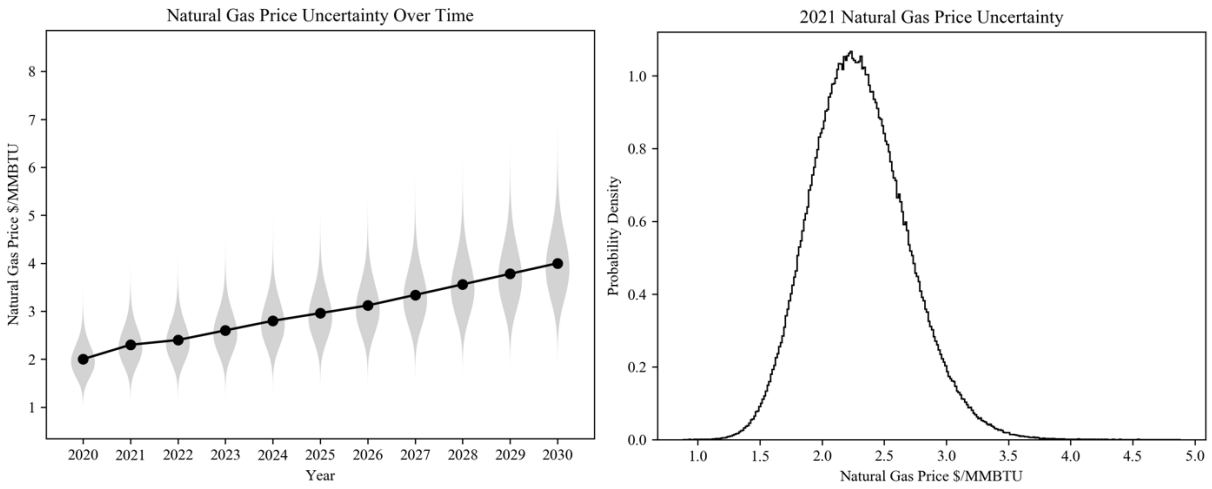

Figure 10 & Figure 11: Natural gas price uncertainty over time & 2021 probability distribution function

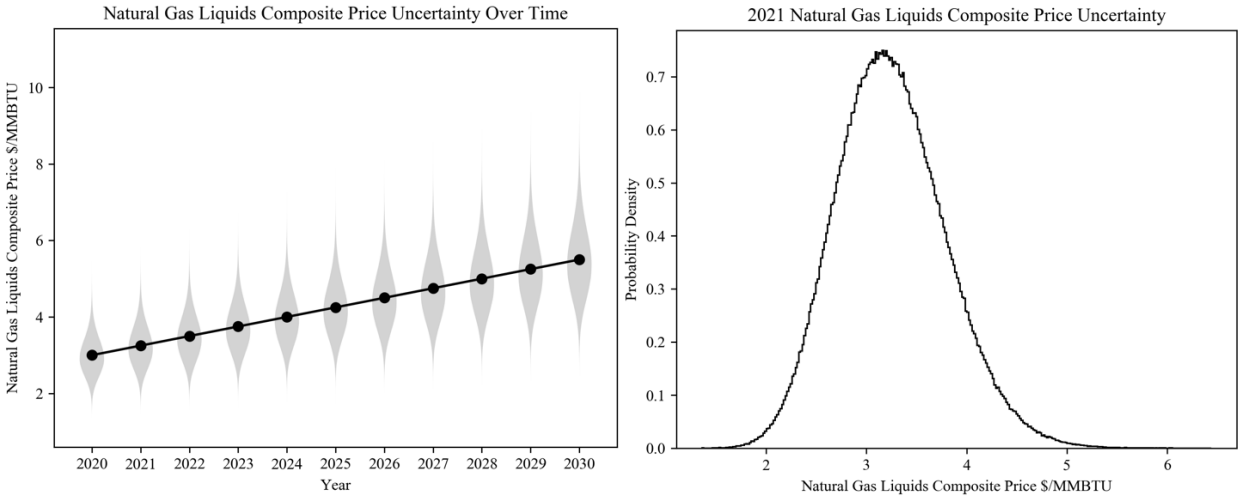

Figure 12 & Figure 13: Natural gas liquids composite price uncertainty over time & 2021 probability distribution function

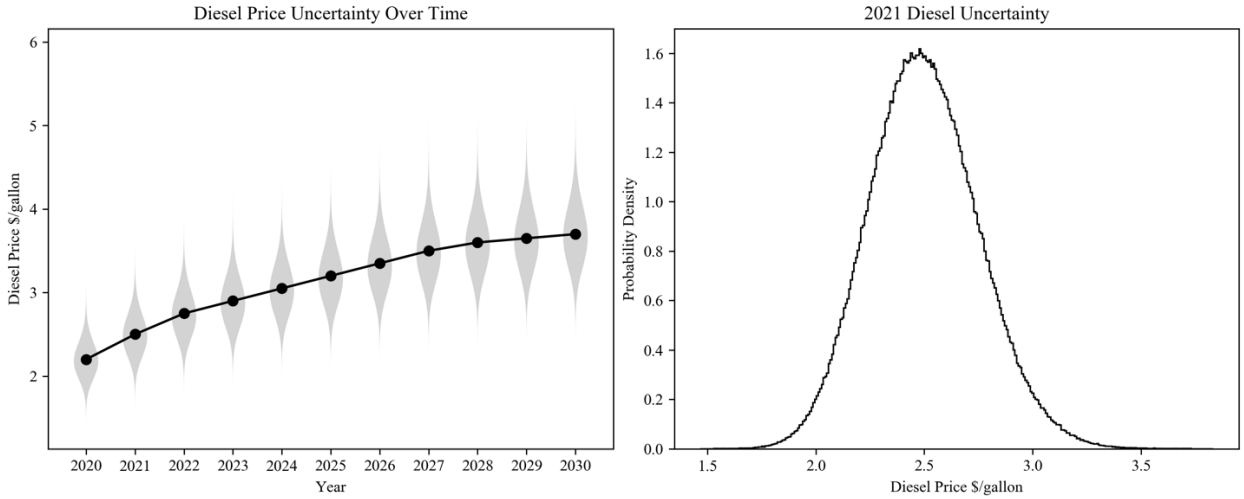

Figure 14 & Figure 15: Diesel price uncertainty over time & 2021 probability distribution function

For each well production subsystem, the annual decline rate, GOR, water cut, and gas heating value are selected from a normal probability distribution function (equation 4). Uncertainty in drilling, fracking and compression days, used to calculate combustion emissions, are modelled with normal distributions. Figure 16 through Figure 21 plot output histograms from calling the corresponding method 1e6 times.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \qquad (4)$$

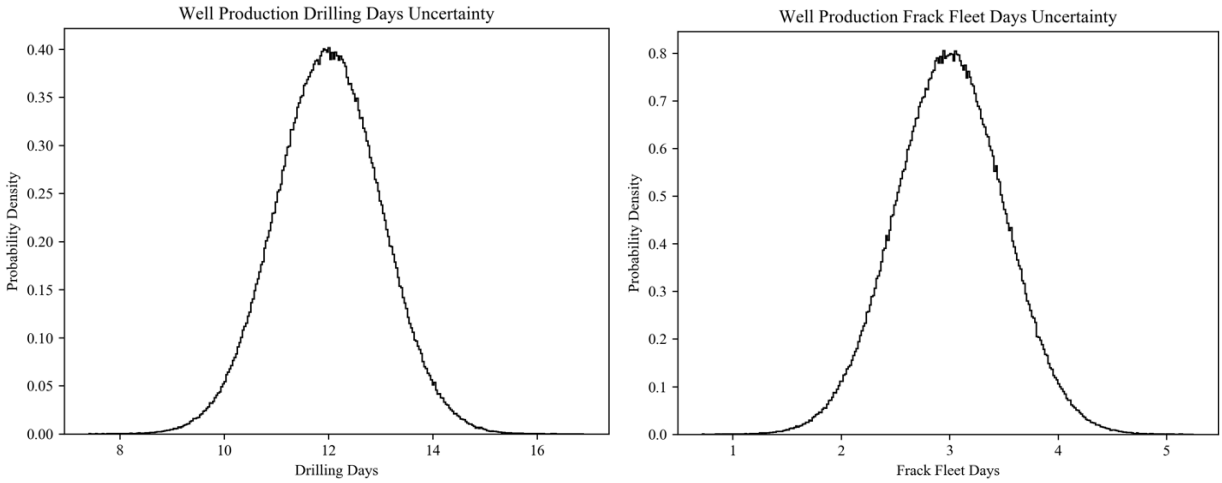

Figure 16 & Figure 17: Drilling & frack fleet days probability distribution functions

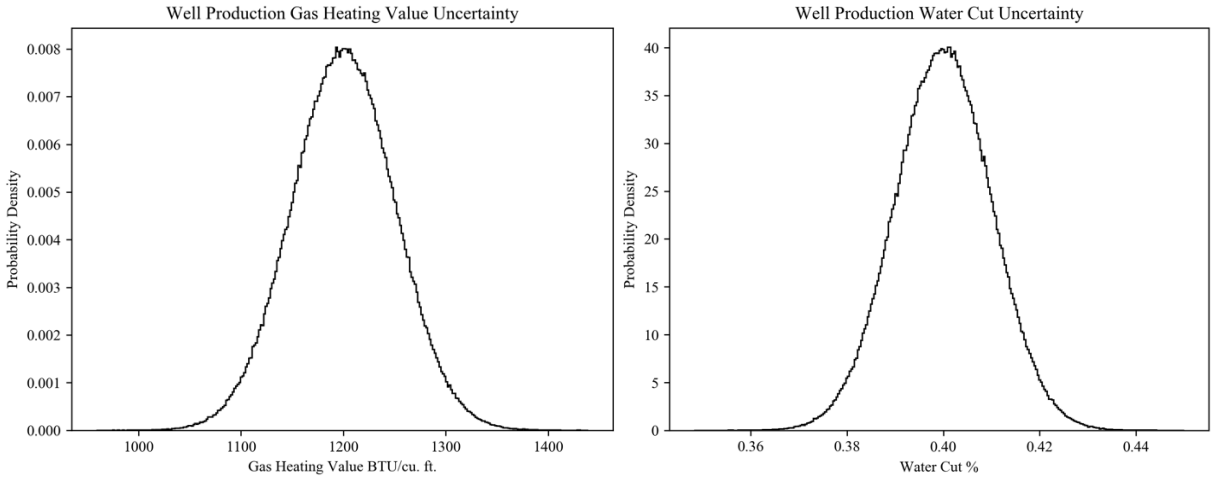

Figure 18 & Figure 19: Gas heating value & water cut probability distribution functions

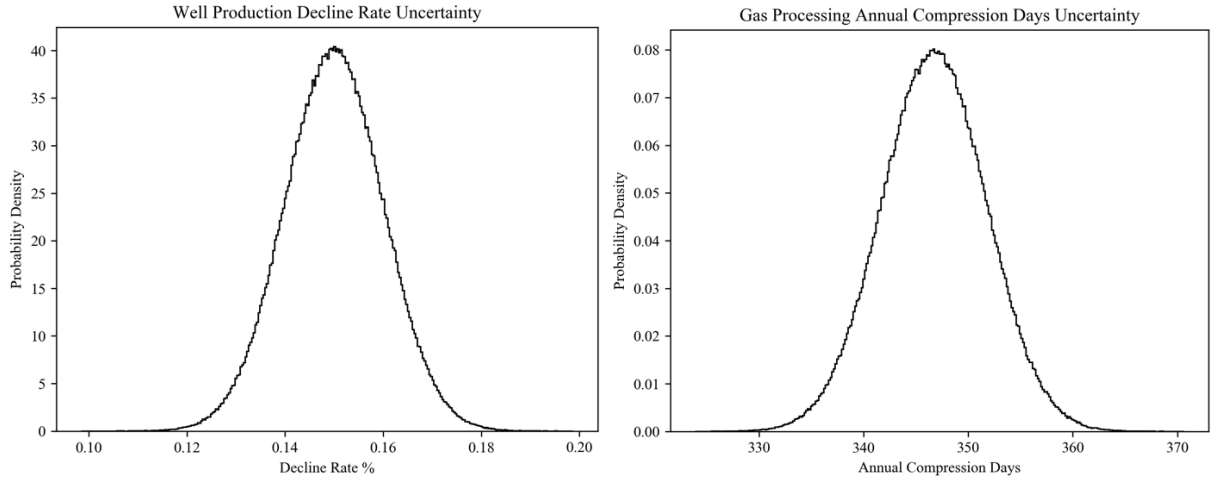

Figure 20 & Figure 21: Decline rate & compression days probability distribution functions

Comparing system architecture strategies

The program output show differences in scenario performance after simulation (3000 runs). Comparing net present value (Figure 22), the planned capacity scenario performs poorly compared to the expansion scenarios. With the planned capacity we constrain the production so if targets increase due to an increase in oil price, the system is capped. This can be seen in the total capital spend (not normalized or discounted) comparison in Figure 23. Additionally, since we assume that the larger subsystems take three years to operationalize, we delay the ability to drill wells and realize positive cash flow.

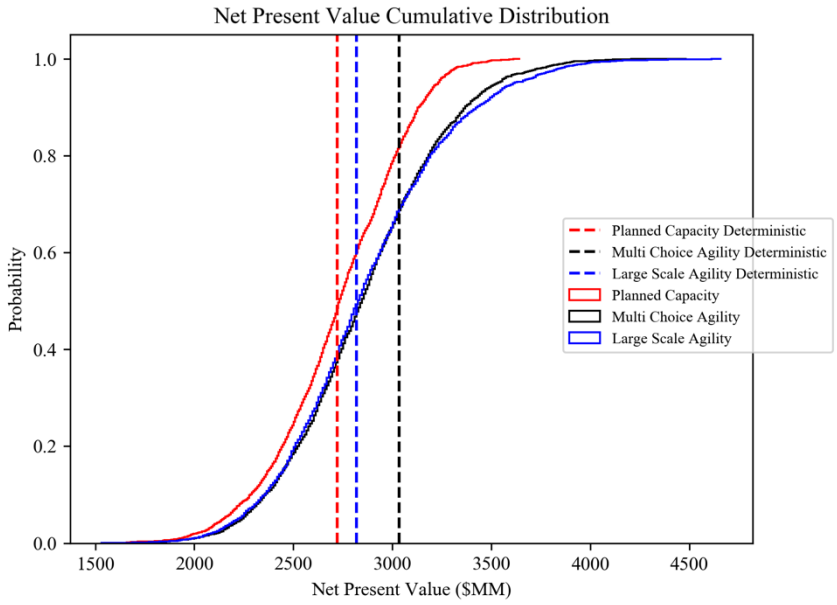

Figure 22: Net Present Value for Initial Scenarios

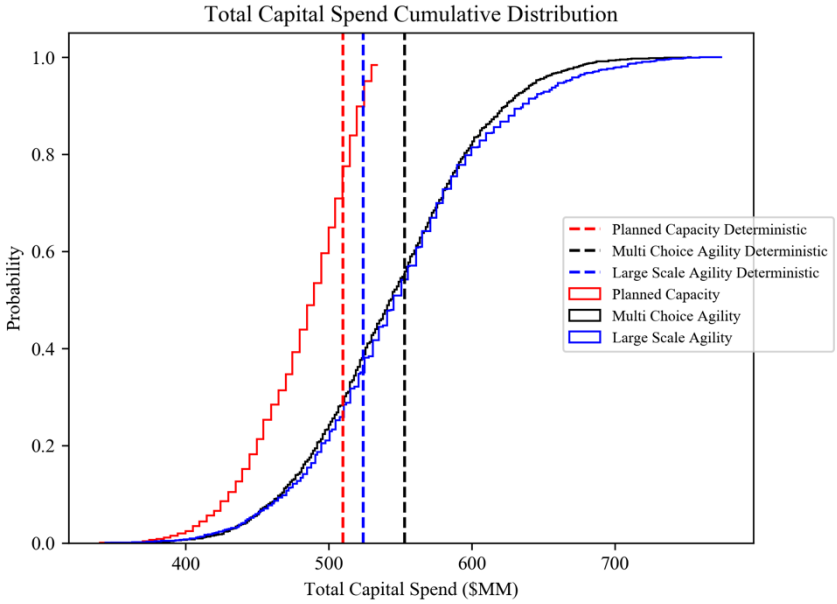

Figure 23: Total Capital Spend for Initial Scenarios

Comparing the expansion scenarios, the large scale agility scenario slightly outperforms (NPV) the multi choice scenario under uncertainty. Interestingly under unchanging conditions, the multi choice system performs significantly better. This is mainly due to the early production years, when production is realized faster since the large scale system waits two year for the large oil and medium gas processing subsystems to come on line.

Further information to support recommendation

The system performance curves (Figure 22 and Figure 23) show some of the information needed to make a recommendation. Reviewing the annual profiles for overall capital, facilities spend and oil production gives additional insights (Figure 24-26). We can also confirm and analyze the total number of subsystem types that were decided using the optimization algorithm (Figure 27). The planned capacity profile has two capital investment peaks, a large facility investment in 2021 and then drilling a large number of wells to fill the facility when it is ready for operation in 2024.

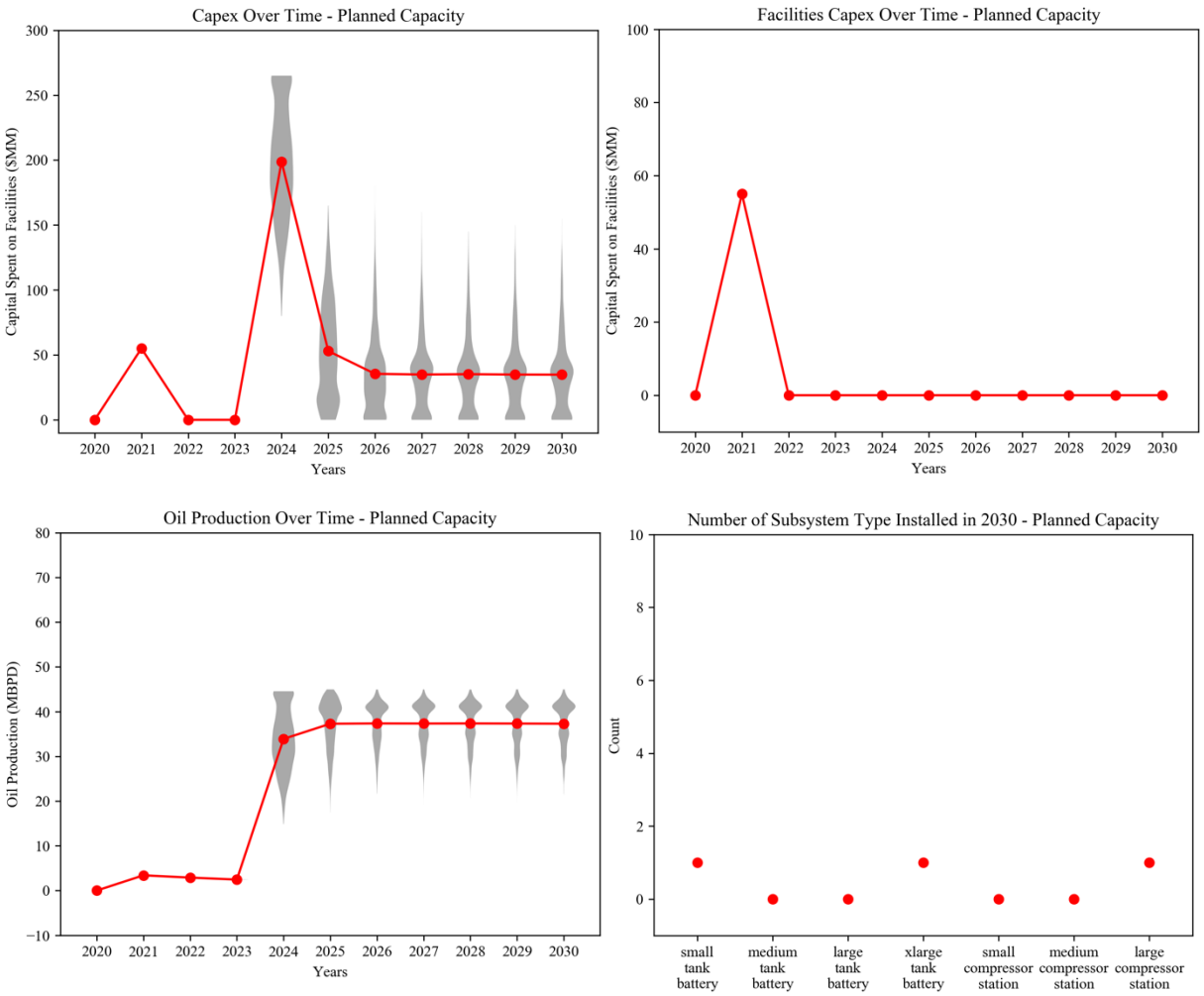

Figure 24, 25, 26 & 27: Capital Spend Profiles, Production Profile & 2030 Facility Subsystem Types – Planned Capacity

Reviewing the investments made for the large scale agility scenario, the majority of the time, three trains, where a train is a large tank battery and medium compressor station, are installed by 2030 (Figure 31). The investment comes in two to three separate peaks (Figure 28).

Most simulation runs build a train in 2021 and 2023 (Figure 29). The third train is built sometime after, depending on when the production target increases due to an increase in oil price. Part of the additional value for this scenario is the earlier oil production realization (Figure 30) since a train operationalizes a year earlier than a full capacity system (Table 5).

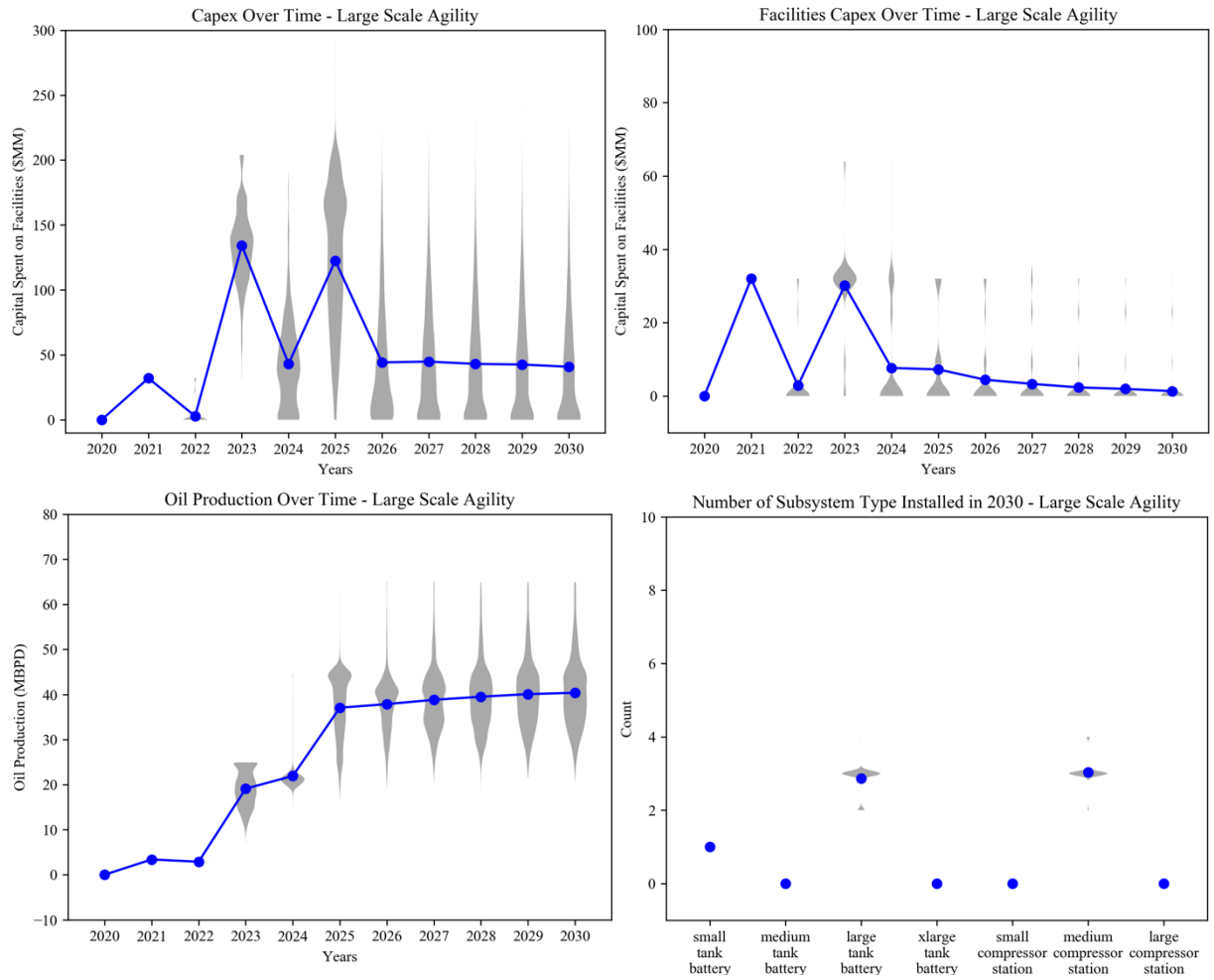

Figure 28, 29, 30 & 31: Capital Spend Profiles, Production Profile & 2030 Facility Subsystem Types – Large Scale Agility

The multi choice agility scenario has a smoother and lower investment profile (Figure 32). The mean spend is less than $100MM and the majority of the facilities spend is spread over the first four years of the system's life (Figure 33). The production profile (Figure 34) is also smoother and closer to the target profile (Figure 36) since facilities operationalize sooner to allow for drilling investments. Interestingly, typically three types of oil processing and two types of gas processing subsystems were selected (Figure 35).

The multi choice system architecture better supports a corporate strategy of agility. The smoother capital investment profile and faster realized production de-risks low likelihood, high

consequence scenarios. Investing smaller amounts in facilities that can be operationalized quickly, can free up capital for other investments in the portfolio.

Additionally, as the system is incrementally developed more information can be used to update the model for future decisions. Learning curves for procurement, construction and design for subsystems installation can be incorporated.

The size of initial investment commitment can affect the timing of decision-making processes. Management may be constrained by annual budgets or larger commitments may need higher levels of endorsement. Looking at the multi choice strategy, to realize any production, in 2021 we need to commit around $20MM for facilities and $30MM in well costs (2022: $50MM total capital minus $20MM facilities. We subtract 2022 facilities cost since their capacity will not be used until they are operational at least a year later).

Making an initial investment commitment of $50MM vs. $150MM (large scale agility) or $255M (planned capacity) may require a different process for management. In addition, we de-risk the investment by seeing production within in a year of the decision vs. two or three years where more can change in the market. A large investment commitment can take longer to endorse and if the decision is delayed changes to the market, technology or system's context may take place during that time.

We've shown a basic analysis for an upstream system designed to incorporate real options vs. deterministic planning. We've compared architectures that allow for expansion based on uncertainty and shown the advantage of preparing and reacting to market changes. The next chapter will explore improvements to the multi choice agility scenario, using the same Python program to refine the subsystem choices.

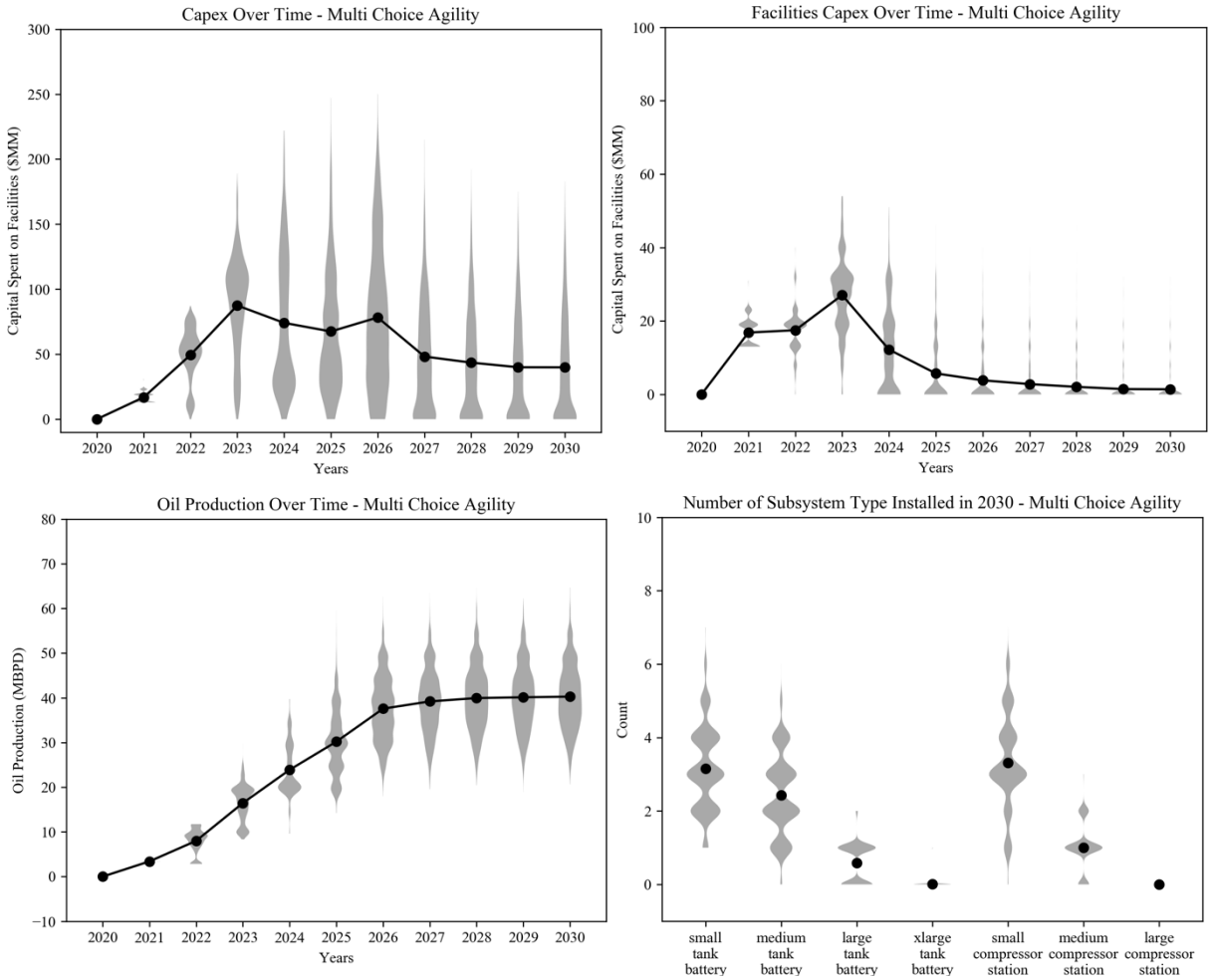

Figure 32, 33, 34 & 35: Capital Spend Profiles, Production Profile & 2030 Facility Subsystem Types – Multi Choice Agility

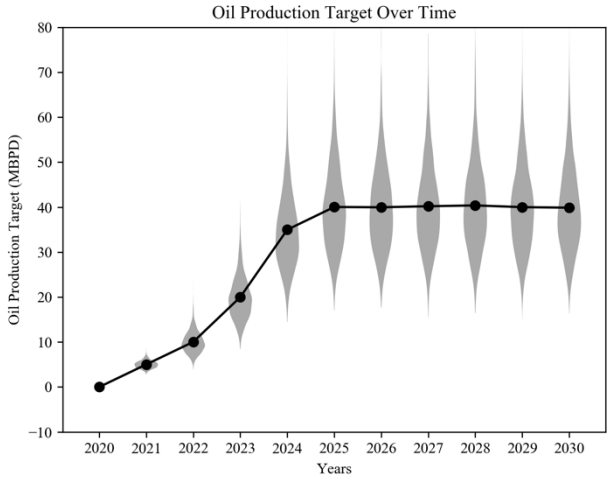

Figure 36: Oil Production Target Profile – All scenarios

Chapter 6: Case Study - Extended Analysis

Refining the subsystem types

In the initial analysis, the subsystem alternatives were broad in the multi choice agility scenario. We gave the program four sizes for oil processing and three sizes for gas processing subsystems. In systems engineering, the definition of subsystem requirements leads to component detailed design (Figure 3). Having many alternatives to continue into the next design and planning phase would be unreasonable and a potential waste of resources. Figure 35 shows which of the types were selected using the optimization program within the simulation. The optimization selects the number and type of subsystem to meet production with the minimal capital investment. At the end of the 10 years, the simulations had selected small, medium and large tank batteries for oil processing and small and medium compressor stations for gas processing.

Refining the agile investment scenario, we eliminate the less selected options of large tank batteries or medium compressor stations. Using the program, the updated scenario can easily be run under the same uncertainty conditions as the initial scenarios. Comparing the system performance is interesting (Figure 37), the refined strategy performs significantly better than both of the original agility strategies. Under deterministic conditions both the original and the refined multi choice strategies have the same performance. This implies that the larger subsystem selections happen during uncertain conditions and it is these choices that create the difference in performance.

Although more capital would be spent over the life of the project (Figure 38), we still see a low but drawn-out capital expenditure profile for the refined case (Figure 39 and Figure 40). The increase in performance can be attributed to facilities becoming operational faster after investment decisions are made. This is evident especially in 2024, where the production average is over 30 MBPD for the refined scenario (Figure 41) vs. close to 20 MBPD in most of the original multi choice scenario (Figure 34). Not only does performance increase but less subsystem types need to be designed (Figure 42), reducing other costs and complexity not included in the analysis.

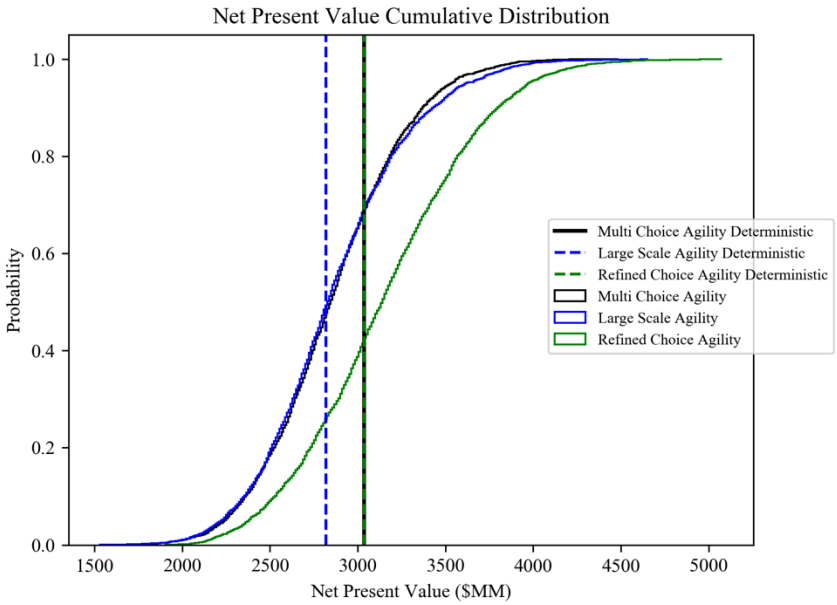

Figure 37: Net Present Value for Agility Scenarios

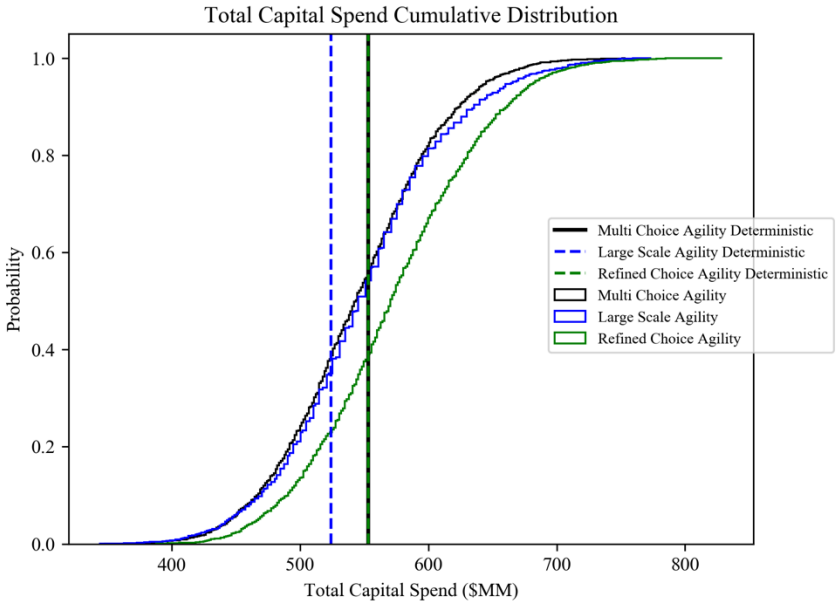

Figure 38: Total Capital Spend for Agility Scenarios

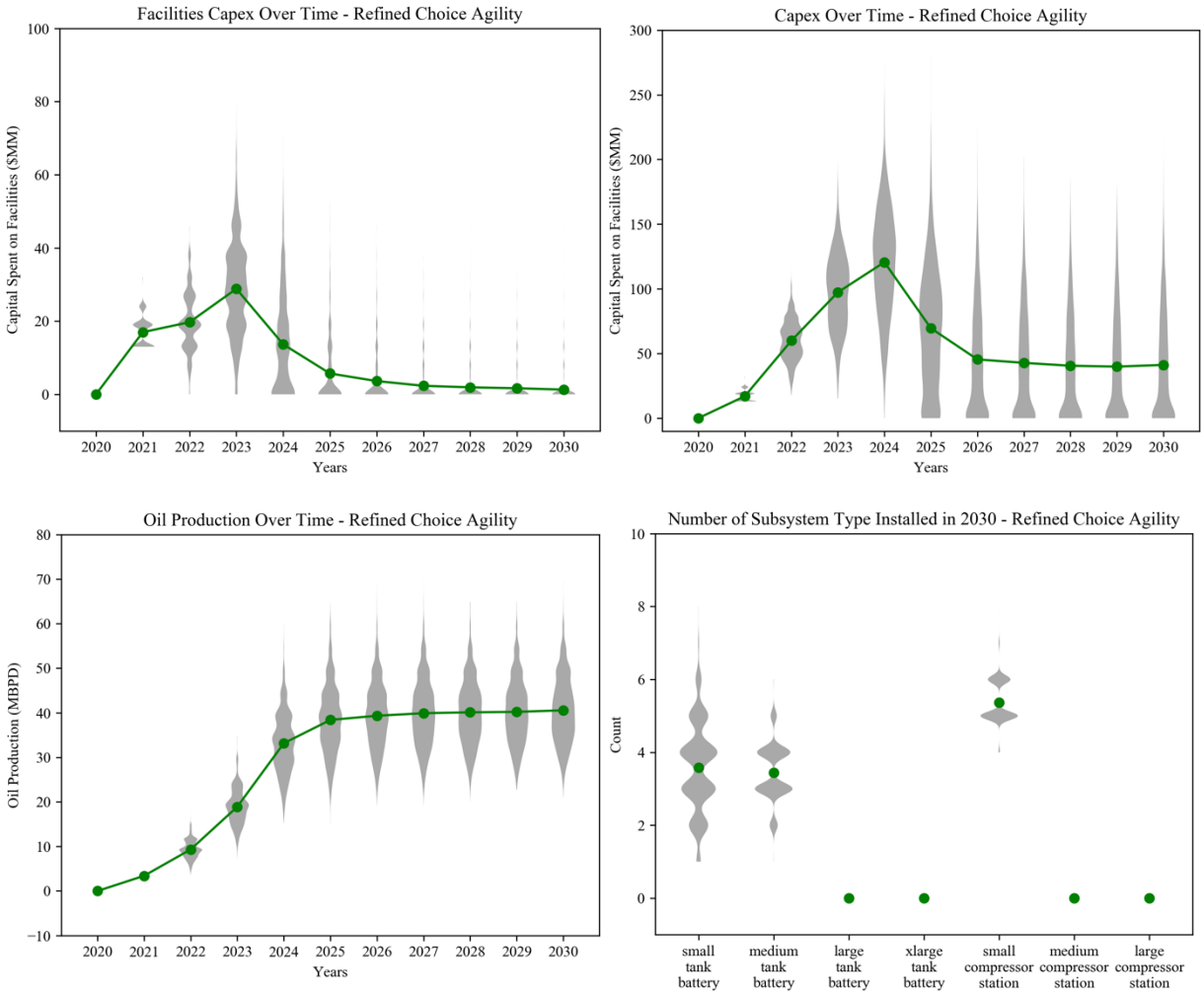

Figure 39, 40, 41& 42: Capital Spend Profiles, Production Profile & 2030 Facility Subsystem Types – Refined Choice Agility

Exploring emissions and energy generation subsystems

We looked at the desired metric of net present value and now investigate the undesirable emergence of emissions (Figure 43). The relationship of the curves is very similar to the NPV curves, this makes sense since production is the main driver of both metrics. For a similar reason the emissions and energy required over time (Figure 44 -51) follow similar profiles to capital expenditure and oil production curves, respectively, for each of the scenarios over time.

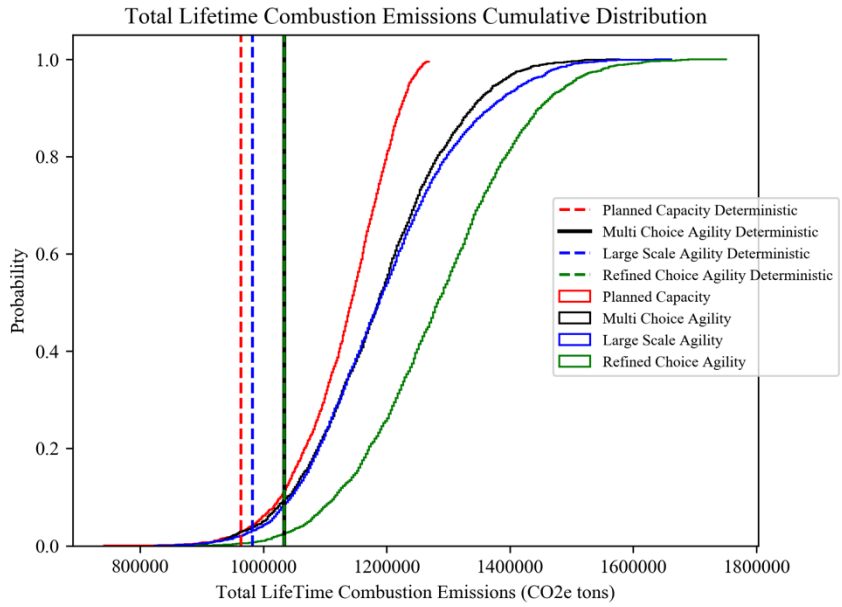

Figure 43: Total Emissions for Uncertainty Scenarios

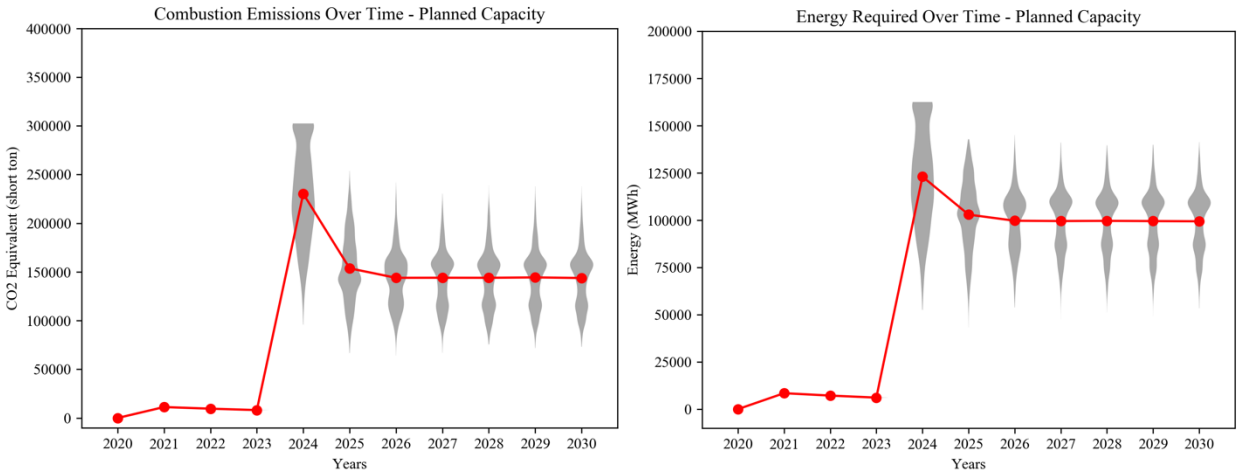

Figure 44 & Figure 45: Emissions and Energy Required Over Time – Planned Capacity

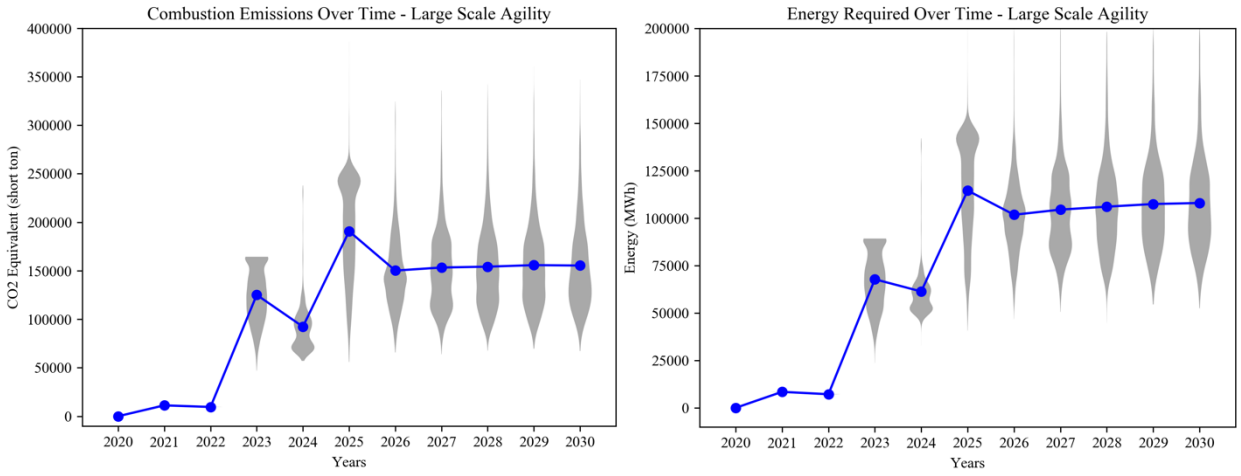

Figure 46 & Figure 47: Emissions and Energy Required Over Time – Large Scale Agility

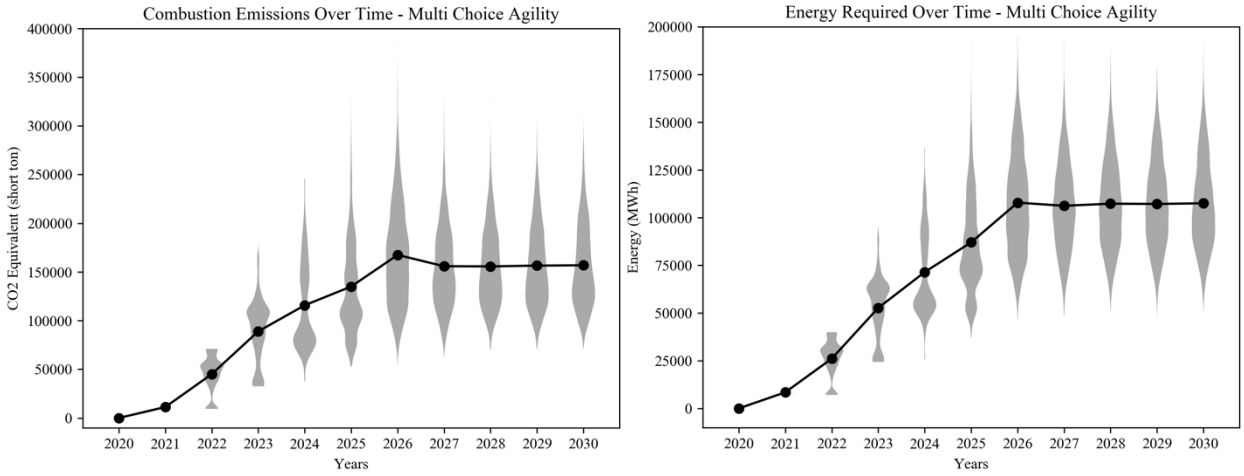

Figure 48 & Figure 49: Emissions and Energy Required Over Time – Multi Choice Agility

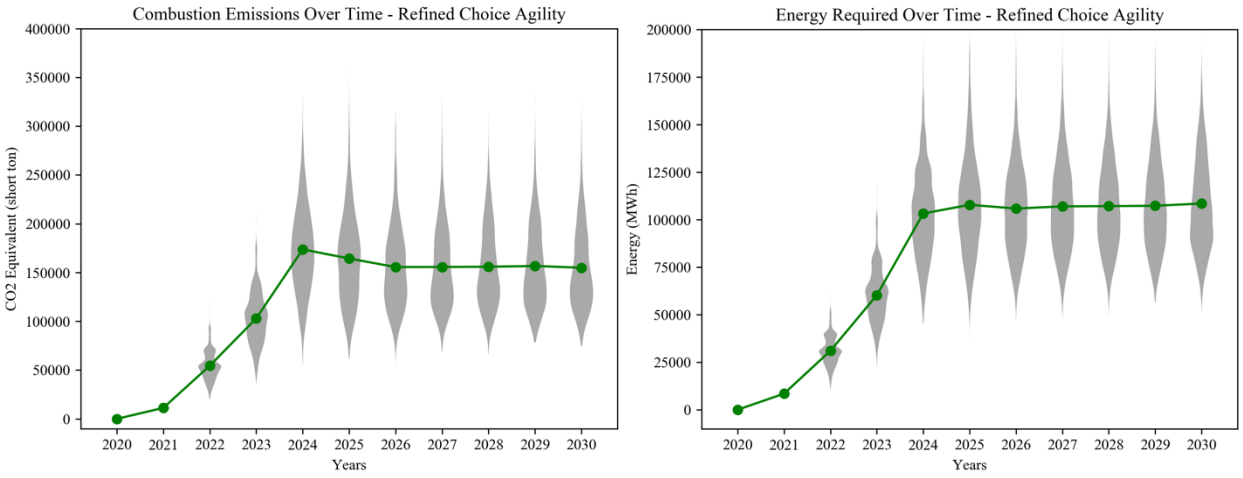

Figure 50 & Figure 51: Emissions and Energy Required Over Time – Refined Choice Agility

In Figure 7 we expanded the system decomposition to include non-traditional energy sources that reduce emissions. We can use the energy required graphs to develop potential strategies to implement these different sources. For the refined choice scenario, annual energy required averages between 100,000 MWh a year for all architectures. This would be equivalent to a 11.4MW power source that has 100% utilization and efficiency. The compression source will have a more consistent demand than the drilling and fracking generators, which may have much higher instantaneous peaks of energy. We will assume for now that these peaks will be supplemented by available diesel sources as in the base case.

To fully support the demand curve, we can estimate the annual power source capacity needed to be operational: 2022: 3.4MW, 2023: +3.4MW, 2024: +4.6MW. To explore the renewable component sources and to keep with a standard size strategy, we compare adding a 3.5MW source for each of these years and the conventional energy sources will make up the difference (Table 6).

	Additional Gas Processing	**Solar**	**Wind**
Energy Capacity Needed (MWh)	30660	30660	30660
Power Capacity Needed (MW)	3.5	3.5	3.5
Factors	0.8 (uptime) 0.3 (efficiency)	0.18 (capacity utilization factor)	0.3 (capacity utilization factor)
Rated Capacity	2 MMSCFD	20 MW	12 MW
Cost	$1MMfor NGL extraction + $1MM for CNG processing	$20MM	$24MM

Table 6: Energy Supply Subsystem Types

After running the updated energy supply scenarios through the program, Figure 52 shows that net present value is very similar for all scenarios. We then look at total capital (not discounted) in Figure 53 and total emissions in Figure 54. As expected, total capital increases significantly for wind and solar farm scenarios and only slightly for gas processing. Also as expected the total combustion emissions is significantly decreased for solar and wind, and less reduced with additional gas processing.

Figure 55 shows total capital and emissions for all 3000 simulations of each scenario, excluding wind. Wind as an energy supply will always have higher capital expenditure than the solar farm scenario. By omitting it in Figure 55, we can clearly see the tradeoffs for the refined

choice scenario with traditional energy supply, with additional gas processing and with solar farm. There is a clear trend, that emissions increase with capital expenditure. This makes sense since additional compression and drilling, especially due to diesel use (Figure 56 and Figure 57) would increase both of metrics. It also is clear that the addition of a solar farm reduces emissions but with a greater increase in capital expenditure than with gas processing.

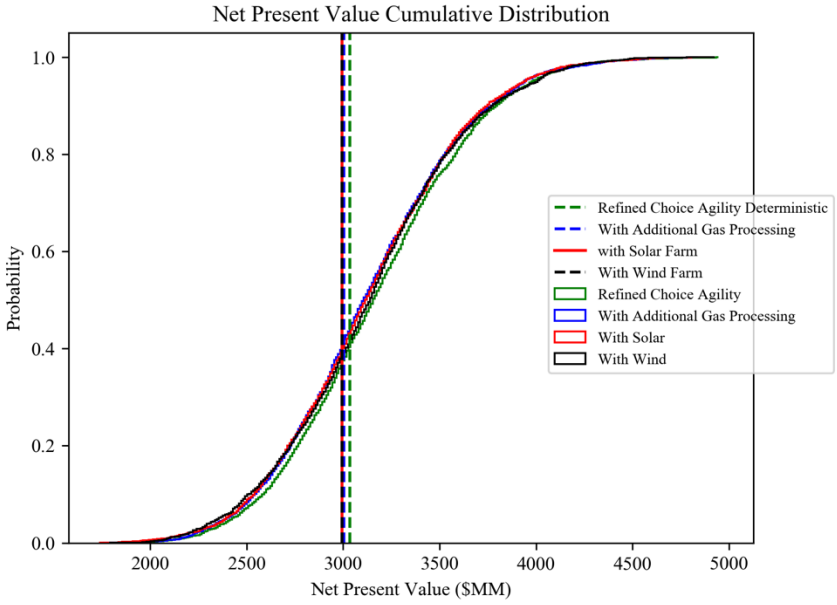

Figure 52: Total Net Present Value for Energy Supply Scenarios

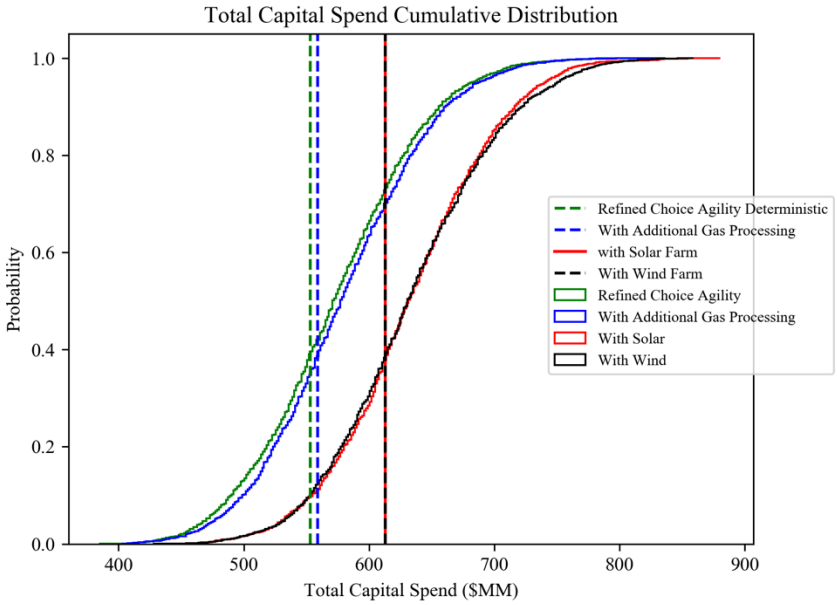

Figure 53: Total Capital Spend for Energy Supply Scenarios

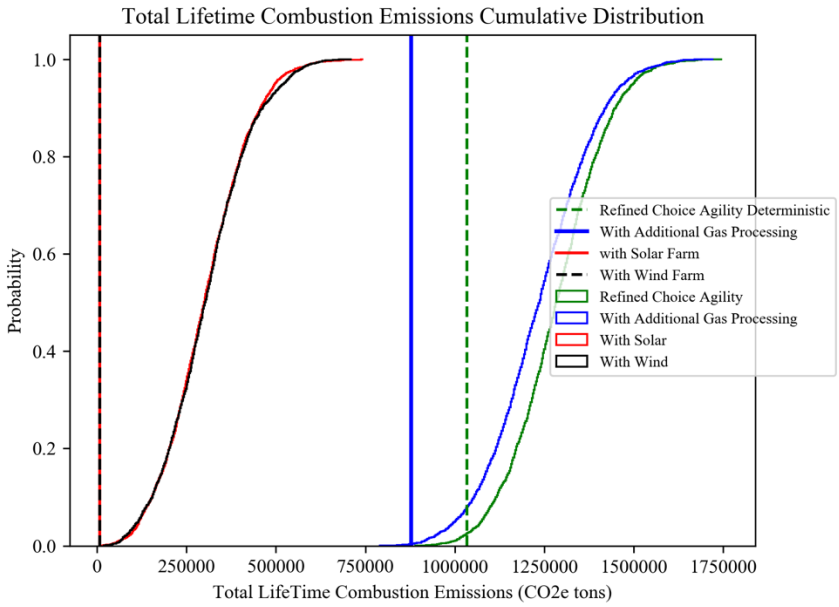

Figure 54: Total Combustion Emissions for Energy Supply Scenarios

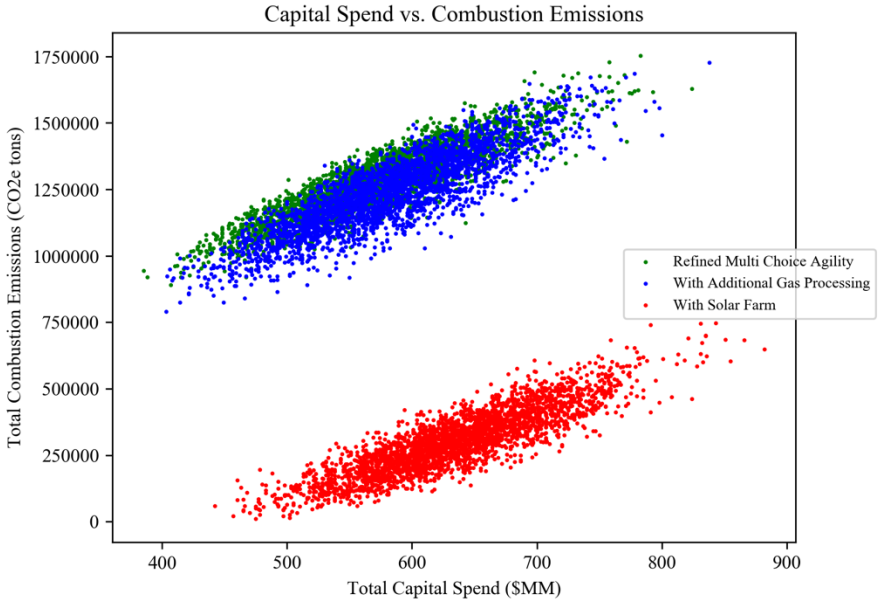

Figure 55: Total Combustion Emissions vs. Total Capital Spend for Energy Supply Scenarios

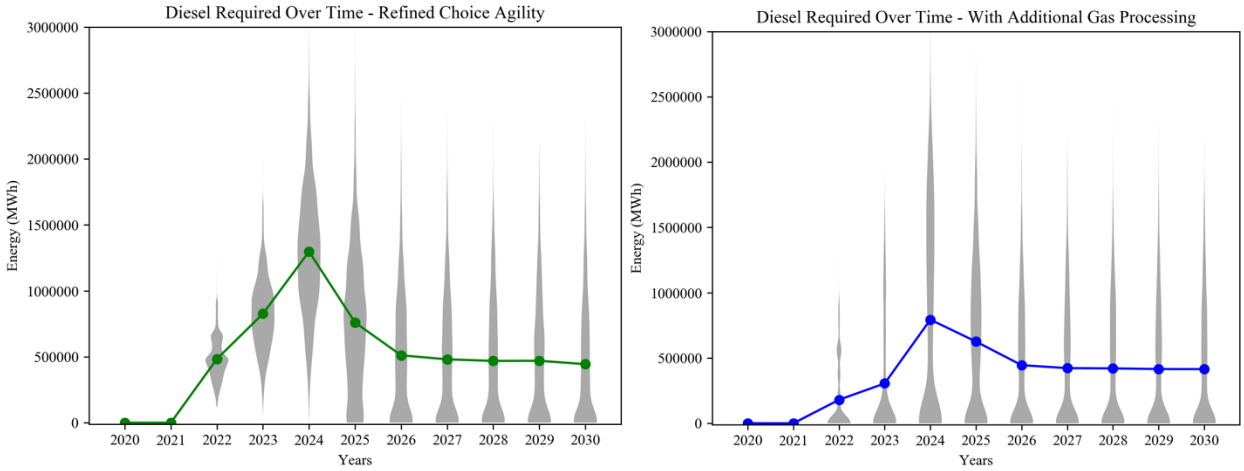

Figure 56 & Figure 57: Diesel Required Over Time – Refined Choice without and with Additional Gas Processing

Subsystem recommendations

Based on the analysis the recommendation is to develop the field using a strategy of investment agility. The subsystems should meet the requirements in Table 7. The initial commitment in 2021 is $60MM, $21MM for facilities (1 medium tank battery, 1 small compressor station with additional gas processing) + ~$39 MM for drilling of wells in 2022 (Figure 58 and 59). Investment decisions should be revisited annually.

The expected net present value is $3.2B (Figure 52), when compared to the initial planned capacity scenario is almost a $500MM increase. Expected lifetime emissions from combustion sources is 1,250,0000 CO_2e tons, a reduction of 75,000 tons for the multi choice scenario. Annual capital expenditure is expected to peak in 3years at $150MM, however in the first two years commitment is under $100MM. Production starts only one year after initial commitment.

The team will need to develop the design, procurement and construction plans for the recommended subsystems and their integration. In addition, the team should study the system changes and costs to use a solar farm as a substitute or supplementary energy supply. Once additional cost information and technical challenges are explored, we can use the same Python program to evaluate the impacts of paying for this flexibility. We are recommending this additional work since the emissions reduction is extremely beneficial, but the cost is high under the initial strategy studied.

A solar farm has the potential to have greater agility than the other subsystems explored, which were assumed to be operational after a year or more. Solar farm infrastructure may take this long to implement, but additional solar panels with smaller capacity could be operational faster. This agility was not explored in the model, but after further engineering, the model could be updated to evaluate the impacts on the system. If necessary, the Python program could be updated with an additional "decision" (optimization algorithm) for how many solar panels to be installed annually.

Innovation in solar energy and battery technology will also continue, with the potential to reduce costs and increase utilization factors. Since the system will be designed for subsystem agility and solar farm flexibility, the model can be used annually to make expansion decisions. The inputs can be updated with lessons learned during operations and market prices for equipment. The strategy allows the business decision maker the option to continue or stop investment, but they can also take advantage of learning curves from previous subsystem installations and other solar farm users.

Subsystem	**Oil Processing**		**Gas Processing**	**Energy Supply**	
Description	Small tank battery	Medium tank battery	Small compressor station	Additional gas processing	Solar Farm
Capacity	5 MBPD	10 MBPD	20 MMSCFD	2 MMSCFD	Continue engineering
Cost ($MM)	8	14	5	2	
Years to Operational	1	1	1	1	

Table 7: Recommended Subsystem Requirements for Further Design Development

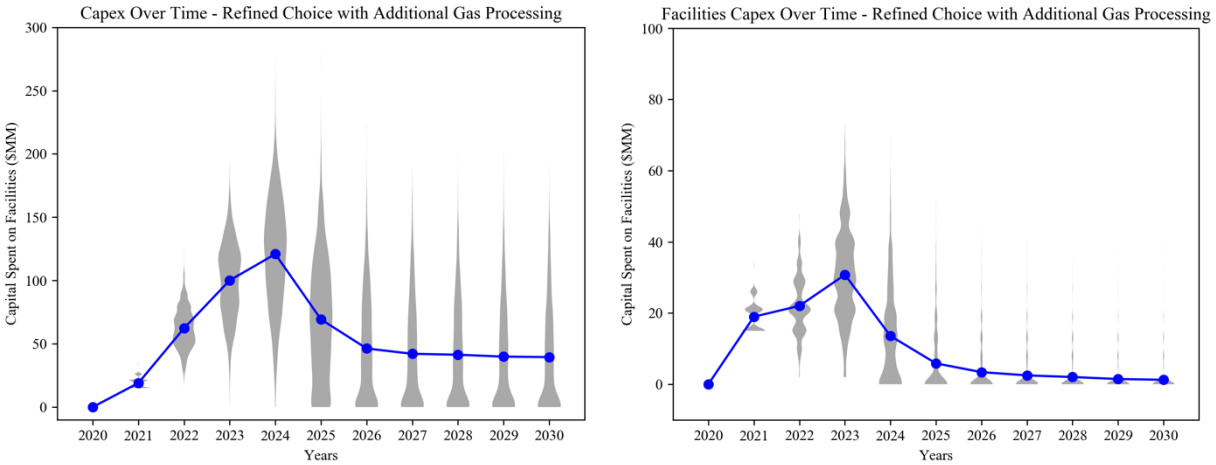

Figure 58 & Figure 59: Capital Spend Profiles for Recommended Design

Chapter 7: Adapting the Model and Future work

Adding and changing subsystems

The model's architecture is one way to decompose an upstream system. The system architecture includes primary value subsystem and then expanded to explore a secondary subsystem for combustion emissions. Different subsystems, especially supporting functions and objects can be added depending on specific use cases. These additions can be done in both the OPD and SysML system models and then as a child class in the Python program.

A deepwater oil and gas development may be another use case. Deepwater fields are located under a body water. The hull is a floating structure that supports the majority of the primary value instruments (processing equipment). The OPD could be expanded to include the hull as a subsystem (Figure 60). An additional child class and its interactions could be added to the program's subsystem class. Additional subsystem attributes needed to evaluate the interaction with the hull are weight and footprint. Architectures with different hull capacities could then be examined. For example: the hull is designed for planned capacity; a larger hull is installed to allow for further expansion of other subsystems; or even a scenario with multiple hull designs. These architectures can be evaluated against uncertainty conditions.

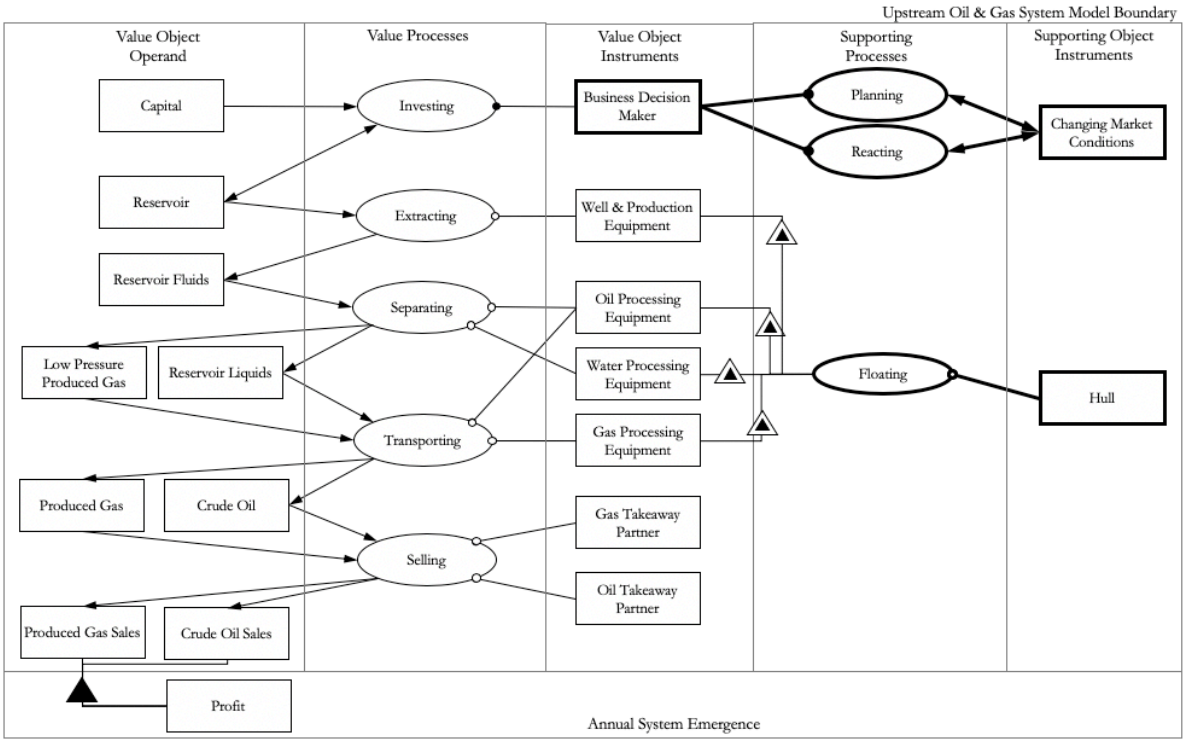

Figure 60: System Architecture Object Process Diagram - Deepwater Asset Type

Adding and changing decision strategies

In the current program there are three main decisions: 1) How market conditions effect production targets, 2) How many wells to drill annually and 3) How many and what type of oil and gas subsystems to build. The development of these decisions stems from three factors, the uncertainty drivers, the decision sequence and the time the decision takes effect on the system.

The decision strategies can also be explored to give insights to different drivers. For example, the relationship between production targets and market prices. Production targets are currently a constraint used in both subsequent capital investment decisions. In the program a simple factor is used to adjust targets. We assume that this decision takes instantaneous effect on the system with no delay in becoming an annual constraint. The simple algorithm used to define the change in production target due to oil price could be derived as an output from a business unit or enterprise-wide optimization program.

The integer optimization program used to simulate subsystem selection could also be modified. By modifying the objective function of the program, decision strategies can be studied. The current program drills wells to maximize the capacity of the system but is constrained by the production target. Instead the objective function and constraints could be written to only be constrained by system capacity. Or the optimization program could be modified with additional constraints like number of rigs available for simultaneous drilling.

The third decision's optimization could also be modified. An example of this is another unconventional asset but we want to explore multiple field developments with a large footprint. The future well locations may be spread over a large surface area, many miles apart. In this case an additional supporting subsystem should be added to the architecture for the transfer of fluids (Figure 61). Long pipelines between the different processing locations could become costly. The integer optimization algorithm could then be converted into a multi-integer linear program (MILP). The MILP's objective function minimizes the capital cost of the subsystems, including pipeline costs. Decision variables would include continuous variables for the volume of oil, water, or gas being transferred between subsystem locations. An example MILP set up is in Appendix B.

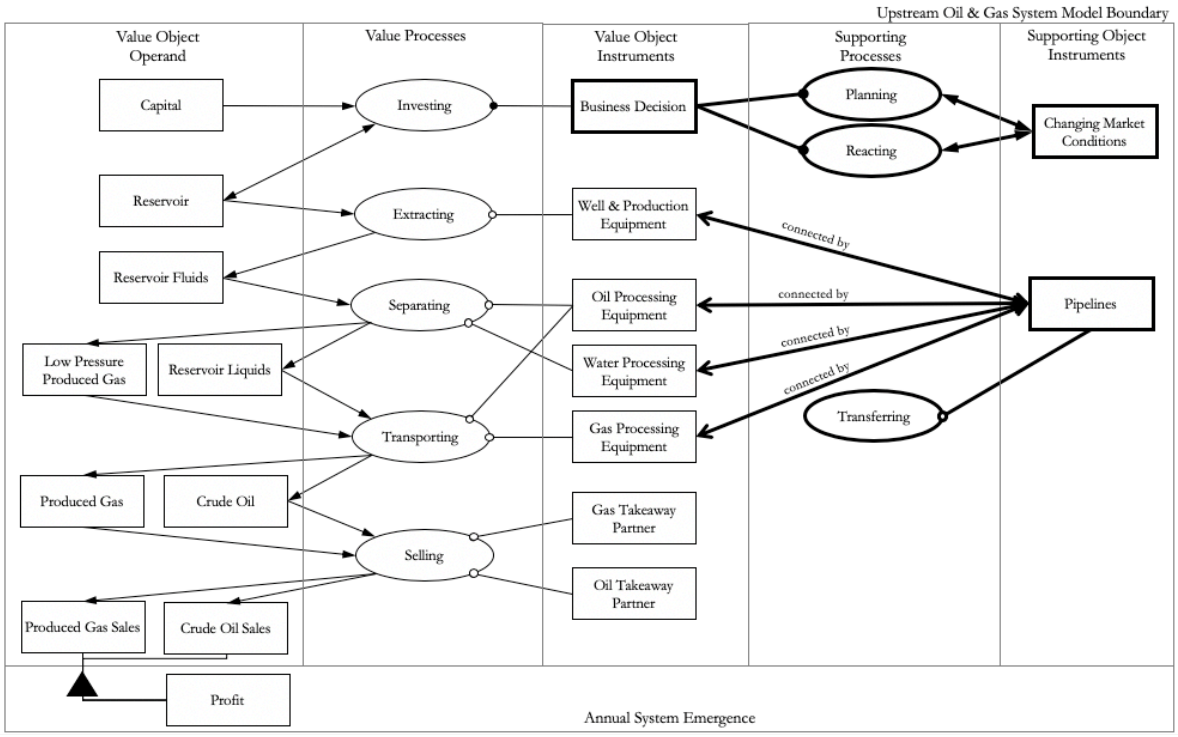

Figure 61: System Architecture Object Process Diagram – Shale Multi-Field Asset Type

Operationalizing the model

We created the program to be efficient using assumptions and simplified calculations. For example, net present value calculations do not take into account taxes, depreciation or additional operating expenses outside of fuel costs. Conversions for fuel to energy output use one rate for generator efficiencies. To be more accurate these would use load efficiency curves. Additional assumptions use simple heating value calculations for NGL and residue gas production. There is industry used simulation software that do a better job of chemical process calculations, like Aspen HYSYS. To use the program in a corporate setting, assumptions would need to be vetted and documented, since opportunities should be compared on a consistent basis.

The program takes ~20 minutes to run 3000 simulations when ran on a local laptop computer. To scale the program's processing efficiency, cloud computing would need to be considered to reduce simulation time. Additionally, we used an academic Gurboi™ license for integer optimization. Other solvers could be assessed for corporate licensing.

A friendly, easy to use user interface (UI) would need to be developed. Although it is simple to manually change inputs in the Python code, the normal user would need an easier way to interface with the program. The user's ability to change inputs, subsystem types, system calculations and output graphics would need to be evaluated and incorporated in the UI.

Chapter 8: Conclusion

We developed a program that demonstrates a practical and efficient method to quantify the tradeoffs between building small with agility vs. large scale. The program allows for expansion decisions made under uncertainty to be included in the early stages of system architecture development. There are business and market conditions when one strategy may outperform the other. The method allowed us to investigate subsystem changes by simulating their effect on overall system performance. We explored architectural decisions and made real, specific recommendations for subsystems requirements.

Projects in oil and gas have a long history of following a bigger is better strategy. Uncertainty in the hydrocarbon market is not new but seems increasingly volatile in the next decades. From increased pressure to confront climate change, cheaper renewable technologies and continued innovation in hydrocarbon production, much seems uncertain. To be competitive under uncertainty, we proposed projects designed for agile investment as a strategy. We suggested leveraging two enablers, annual decision making and smaller, quicker to operation standard subsystem designs. To quantitatively compare strategies, we combined and expanded concepts in system architecture, real options analysis and object-oriented programming.

Starting with system architecture modeling, this step was needed for communicating and defining the system to be simulated and analyzed. Using the modeling languages of OPM and SysML, along with the techniques of abstraction and zooming, these diagrams paint a clear system framework. We applied these tools to an upstream, oil and gas project. Decomposing the oilfield's production system into seven physical subsystems.

Using the framework and concept diagrams, the system architecture was translated into an object-oriented programming language to quantitatively evaluate the architecture. Parent and child classes for system lifecycle, annual system and subsystems were developed from objects defined in concept modeling. Class methods and functions mirror the processes and interactions found in the concept modeling. The program first analyzed scenarios using deterministic conditions and then uses Monte Carlo simulation to evaluate under uncertain conditions.

Variables with uncertainty are modelled using probability distribution functions and the program runs simulations by discretely choosing inputs from these distributions. Running of the program emulates system development over time and the emergent metrics are shown in a cumulative distribution graph. The graphs indicates the probability of the emergent outcome. Using object-orient programming allows for the program to store class instances and their

attributes as the simulations run, allowing for further exploration and analysis of the outputs and decisions made.

To evaluate the real options of expansion, the program must simulate the decisions made under changing and uncertain conditions. This was done using integer optimization programs embedded into the simulation. The output gives the optimal investment decision during that point in time given the conditions known at the time to the decision maker. This is powerful as decision requirements could also be changed to examine different decision strategies' impact on system performance.

To show practicality of the program, we used it to study a hypothetical, but representative example. The development of a small, onshore, unconventional oilfield would need all of the subsystems defined in concept modeling. The development could be done as a large scale, one-time investment or with varying levels of investment over time. In the analysis we first show overall value in agile investing with an expected 5% increase in net present value. The upper tail of the cumulative distribution curve is the most insightful, we see how planning to build only once, to a predetermined capacity, caps potential value if the market price increases. Additionally, we see the value in getting production online faster, in the early years when there may be less uncertainty and revenue is not as discounted.

Further investigation of the agile investment strategy leads to a refinement in subsystems, which actually increases net present value again, over 18% vs. the system with planned capacity. We also explore non-traditional energy sources, which can be leveraged since the capital profiles are more consistent. With this strategy the drilling profile and therefore diesel use is smoother over time. We look at replacing the diesel, with lower emission fuel sources. Recommending additional gas processing to be implemented over several years reduces lifetime combustion emissions by 6%.

In this case we justify the agile investment strategy not only because of performance but also because of the committed capital profile. In the planned capacity, to see any returns from production we must wait 4 years with an investment of $255MM. With our recommendation, 2021 capital needed is $60MM, with the ability to revisit the next commitment the following year. The recommendation does increase the number of subsystem types to be developed, but this number was reduced and refined from an initial multichoice proposal. Keeping in mind that this analysis would be completed in the early phases of system design, we would want to balance

agility choice with the work required to develop those choices. Design, procurement and construction planning would need to be completed for each subsystem type.

For this thesis these steps were completed by the author but in reality, different teams and domains would complete them. The importance of communicating between these steps is paramount to support decision making. Not only does the complexity of a system need to be communicated but the quantitative analysis' logic needs to be understandable and emulate the architecture.

The system analyzed is complex and complicated. Typical systems have hundreds, if not thousands of components and physical pieces of equipment with many types of processes, interactions and relationships. Analyzing just one of the subsystems or components of the system is the work of teams of subject matter experts, such as geologists, drilling engineers, surface facility engineers, electrical engineers. Cutting through the complexity, business decisions and architectural development strategies must be developed to give clear direction for the detailed work and engineering of the subsystems done by the varying teams.

Future work on the program to address high level simplifications and omitted cost assumptions is needed. Additional subsystems, decisions and metrics could also be added. Using the program requires coding experience in Python so user friendly interface should be developed. Visuals were created in Python, but refinement or other visualization software could be used to interrogate the outcomes.

Decisions must be made in the face of this complexity. Combining the right level of abstraction but with the right level of detail for quantitative insights is a challenge. The upstream energy system itself is complex and so is its context, both the system and the conditions it operates in have uncertainties.

This work expands the fields of real option valuation and system architecture by combining them with object-oriented programming. Using concepts and tools from all three fields, a complex system's performance was simulated. By using the program and repeatedly refining the inputs, persuasive arguments can be explained for specific subsystem requirements. We've developed and demonstrated a method to analyze subsystems designed for agility in the early phases of system development.

Appendix A: Python Program Detailed Explanation

The Python program used to evaluate system architecture implementation scenarios contains 8 modules (about 1600 lines of code). In the next sections we provide descriptions of the modules and example code.

`main.py`

To start a system analysis there are three main inputs into the program, an initial system containing existing and potential subsystems, potential facility subsystem types and production targets for each year of the evaluation timeframe. The initial system is the first index of a list of annual system instances. This list will be built upon to create the evolution of the system throughout its lifecycle.

The production targets are a Python dictionary class with the key as a year and value of oil production for that year. The program first evaluates the system lifecycle deterministically and the production targets do not change. However, when the program continues into simulating uncertainty these targets change on an annually based on a changing oil sales price, meant to simulate the uncertainty in market conditions. The oil production target for the current year and next year is updated to reflect a scaled change for the realized oil price vs. the forecasted expected price.

The main.py module houses the code that starts the program and calls the other python modules to execute their code. The final output is a simulated system over the entire lifecycle. The system lifecycle attributes of net present value and total lifecycle emissions are then calculated and plotted. Additionally, every simulation object is stored allowing details of the annual decisions and attributes to be further explored.

`systemClasses.py`

There are three parent classes used to describe the domain: SystemLifecycle, System, and SubSystem, this emulates the first three levels of decomposition abstraction discovered in the formation of Figure 7.

The System Lifecycle class have the attributes name, description, evaluation year, discount rate and a list of annual system object instances. It also has two methods that can be called to evaluate the system lifecycle using its attributes, creating net present value and total

emissions. The SystemLifecycle class definition and methods is shown below as example of the Python syntax for class creation.

```
class SystemLifecycle():
    def __init__(self, name, description, evalYear,
systemAnnualInstances, discountRate = 0.15, netPresentValue=None,
totalEmissions=None,):
        self.name = name
        self.description = description
        self.evalYear = evalYear
        self.discountRate = discountRate
        self.systemAnnualInstances = systemAnnualInstances
    def evaluateNPVNoCarbonPricing(self):
        npv = 0
        for s in self.systemAnnualInstances:
            if s.year >= self.evalYear:
                t = s.year - self.evalYear
                cashFlow = s.productionRevenue - (s.capexDrilling
+ s.capexFacilities + s.fuelCosts)
                npv = npv + cashFlow/((1+self.discountRate)**t)
        self.netPresentValue = npv
    def evaluateTotalEmissions(self):
        totalLifetimeEmissions = 0
        for s in self.systemAnnualInstances:
            totalLifetimeEmissions = totalLifetimeEmissions +
s.emissions
        self.totalEmissions = totalLifetimeEmissions
```

The System class is used to describe the system as an annual snapshot in time. The object class has several attributes, including name, description and the year of the system instance. Similar to the SystemLifecycle class it has a list of decomposition objects, in this case SubSystem objects. Additional attributes are used to record values that were used during the creation of the object instance or were created to evaluate the system. Attributes for the market conditions (pricing), the oil production target and realized production are recorded for traceability during an annual System instance creation. Other attributes like capital, fuel costs, energy generated, and emissions are evaluated from the class methods. These evaluation methods are called after the subsystem object list is generated.

The SubSystem class defines attributes and methods that all child SubSystems have in common. These attributes include the name, type description, capacity of the subsystem, the units for the capacity, the capital cost, a location index and the year the subsystem was installed. The SubSystem class has child classes which inherit its attributes and methods. The child classes

are created since different subsystems have additional specific attributes or methods that are needed in computation.

The SubSystem parent class has three methods that allow for filtering of a list of SubSystem objects. These methods are used throughout the rest of the program's modules. An example is below. This method provides a filtered list of attribute values based on the subsystem and year installed.

```
    @staticmethod
    def getAttributeOfSubSystemInstalled
(attribute,subSystemList, subSystem):
        output = []
        for ss in subSystemList:
            if ss.yearInstall != 'future':
                if ss.subSystem == subSystem:
                    output.append(getattr(ss,attribute))
        return output
```

Child classes of SubSystem correspond to those shown in Figure 7, WellProductionSubSystem, GasProcessingSubSystem, GasTakeawaySubSystem, OilProcessingSubSystem, WaterProcessingSubSystem and EnergySupplySystem. Defining these as child classes allow them to inherit the attributes and methods of the parent but allow for further flexibility in defining the subsystem.

The WellProductionSubSystem class has the most additional attributes and methods needed to describe the subsystem. This subsystem defines the throughput production in each of the subsequent subsystems. In the program the currentProduction attribute is oil production with unit of MBPD (thousands of barrels per day).

Over time the currentProduction declines and a decline rate or a percentage of oil production decrease is used. For this program a decline rate function (static method) is called in other parts of the program and gives either a deterministic decline rate or a random value from a normal probability distribution describing a probability of a decline rate. The below is example code for returning a deterministic decline rate of 15%. If the seed variable is given this indicates the program is running a simulation with uncertainty and should return a random value from a normal distribution with a mean of 15% and standard deviation of 1%. This is the typical structure for simulating a variable with uncertainty.

```
@staticmethod
def getDeclineRate (seed = None):
    if seed is None:
        declineRate = .15
    if seed is not None:
        declineRate = np.random.normal(.15, .01)
    return declineRate
```

It is common to calculate the gas production using a gas to oil ratio (GOR). The gas production is the amount of gas that must be processed in the GasProcessingSubsystem objects. Water production is usually described by water cut or percentage of a well's total fluid that is water. This factor is used by the program to calculate the water production needed to be processed by the WaterProductionSubSystem objects. Both GOR and watercut are attributes that are estimated using similar methods as decline rate.

WellProductionSubSystem objects contain two of the main sources of combustion greenhouse gas emissions, from drilling and fracking. These methods return the energy required in MWh and the diesel equivalent required in gallons. A normal probability distribution is used for the days spent drilling or fracking and a nominal diesel per day is used. Diesel is converted into energy units using a given diesel generator efficiency.

The other child subsystem classes have additional attributes and methods that are particular to calculations in the program. Using class structure in object-oriented programming is a clean way to organize specifics attributes and behaviors of a subsystem, system and system lifecycle. By using this structure class definitions could be expanded or refined by the subsystem experts and engineers with specialized knowledge.

`priceFroecast.py`

The commodity prices used in oil and gas project evaluations are long term forecasts, decades long. The program uses an annual price forecasts for crude oil, natural gas, natural gas liquids and diesel for calculation of profits and fuel costs. For this case the uncertainty of each commodity price is represented as a normal distribution, with the expected annual price as the mean and a percentage of that mean as the standard deviation.

An example of the oil price definition as a Python dictionary class, the key being a year and value corresponding to a price. The function is similar to the getDeclineRate method in the WellProductionSubSystem class. The function is used in the annualSystemDecisions,py module.

```
oilPriceP50 = {2020:35.00 , 2021: 42.00, 2022: 44.50, 2023: 47.0,
2024: 49.80, 2025: 52.70, 2026: 56.16, 2027: 59.62, 2028: 63.08,
2029: 66.54, 2030: 70, 2031: 73.46}
def getOilPrice(year,seed = None):
    p50price = oilPriceP50[year]
    if seed is None:
        price = p50price
    if seed is not None:
        price = np.random.gamma((p50price/(p50price/6))**2,
(p50price/6)**2/p50price)
    return price
```

annualSystemDecisions.py

The cornerstone of the analysis program is simulating the system annually, taking in the inputs for the year and making decisions that alter the system. The following year the same decisions must be made with new inputs and so on.

Annually there are three major changes, two are driven by uncertainty and one driven by the decisions made in the previous year. The changing conditions for price due to market uncertainty and current production due to decline rate, GOR and watercut uncertainty are simulated using the systemClasses.py and priceForecast.py methods as described earlier. The system itself also changes based on the previous year's capital investments of wells and other subsystems installed.

The annualSystemDecisions.py's function appends an annual system instance object to a list after capturing and "deciding" what changes to the system should be made. The function is recursive, it calls itself after executing, which simulates starting the decision process over again with the changes that were made the previous year.

The function continues to create an annual system instance for each year a production target is available and ends when there is no target available. It first copies the initial system and appends it to the list, updating the year attribute by one. Thus, creating a new annual system to be updated and allowing for the existing subsystem objects to be maintained in the subsystem list.

```
def createAnnualInstances(systemAnnualInstanceList
, subSystemTypes, productionTargets,seed=None):

     #if the last system in systemAnnualInstanceList has the
same year as the last year of the productionTargets end the
function
```

```
        #else capture uncertain conditions and make decisions that
    change the system as a new annual system to append to the
    systemAnnualInstanceList.

        #Function calls itself after making changes
        createAnnualInstances(systemAnnualInstanceList,
        subSystemTypes, copyOfProductionTargets,seed)
```

The beginning of the function body can be thought of as the start of the planning cycle, starting with an existing system infrastructure. Next, the current commodity prices attributes of the system are updated, and existing production is updated using a decline rate.

The changed prices are eventually used to evaluate the system, but also are used to imitate a business decision updating the annual oil production target. In this program the relationship between oil production target and price is simple and calculated by a scaled linear change factor for the realized oil price vs. the forecasted expected price. Both the current year and next year's oil production targets are scaled to be used in the subsequent decision-making algorithms.

`wellsToDrill.py`

Using the updated attributes, the first of two decision functions are executed. The first business decision is how many additional WellProductionSubSystem Objects to install. The production of these new wells will be realized this year, therefore is constrained by the existing system's processing and takeaway subsystems' capacities. The updated current year's production target also acts as a constraint. In the program the decision's objective is to maximize the production of the system given these constraints. A simple integer optimization program was used to determine the optimal amount of production. The equation described was translated into Python syntax using a Gurobi™ solver package extension and license.

Sets

$z = 1, \ldots, Z$: set of future WellProductionSubSystem Locations

Parameters

A: Annual production target

gor: projectd future WellProductionSubsystem GOR

wc: projectd future WellProductionSubsystem watercut

m_z: future WellProductionSubSystem estimated initial rate of production (Barrels per Day)

P: Aggregated current oil production of current WellProduction objects,

G: Aggregated current gas production of current WellProduction objects, calculated using P and GOR object attributes

W: Aggregated current water production of current WellProduction objects, calaculated using P and watercut object attributes

MP : Aggregated capacity of currently installed OilProcessingSubSystem objects

MW: Aggregated capacity of currently installed WaterProcessingSubSystem objects

MG : Aggregated capacity of currently installed GasProcessingSubSystem objects

MT: Aggregated capacity of currently installed OilTakeawaySubSystem objects

MP: Aggregated capacity of currently installed GasTakeawaySubSystem objects

Decision variables

$d_z = \begin{cases} 1 \\ 0 \end{cases}$ Drill WellProductionSystem location or not

Objective Function

Maximize new well oil production

$$\sum_{z=1}^{Z} b_z * m_z$$

Subject to:

$$\sum_{z=1}^{Z} b_z * m_z + P \leq A,$$

$$\sum_{z=1}^{Z} b_z * m_z + P \leq MP$$

$$\sum_{z=1}^{Z} b_z * m_z + P \leq MS$$

$$(\sum_{z=1}^{Z} b_z * m_z) * w + W \leq MW$$

$$(\sum_{z=1}^{Z} b_z * m_z) * gor + G \leq MG$$

$$(\sum_{z=1}^{Z} b_z * m_z) * wc + G \leq MP$$

After the integer program is executed, it will return the future WellProductionSubSystem objects to drill. These are updated in the current system object's list of subsystems which allows the additional production to be captured. The new production needs to be accounted for in the second decision which determines the type and number of the facility subsystems to be installed during the current year.

`facilitiesToInstall.py`

Facility subsystems are assumed to take a year to operationalize, so the decision taken to build is based on the next year's production target, estimated based on the change in current change in price. Similarly, to the well production decision the number of subsystems is to be determined, representing decision making. In addition to the quantity, the type of oil processing and gas processing subsystem are decided.

The assumption is there are multiple standard size subsystems that can be installed at the same potential oil or gas processing location. The subsystem types are given to the function createAnnualInstances when it is called. The subsystem types are created as oilProcessingSubSystem and gasProcessingSubsystem objects with differing capital cost and capacity attributes.

The objective chosen for simulating this decision is minimizing capital spend on the facilities. The decision is constrained by meeting the projected next year's production target, the existing infrastructure's capacities and an estimate of production at year end.

Similar to the wellsToDrill.py, the facilitiesToInstall.py module contains an integer optimization function. The output of this function are binary decision variables for executing contracts for takeaway or installing processing facilities type-location combinations.

Sets

$z = 1, \dots, Z$: set of future WellProductionSubSystem Locations

$bT = 1, \dots, BT$: set of OilProcessingSubSystem Types

$bL = 1, \dots, BL$: set of future OilProcessing Locations

$gT = 1, \dots, GT$: set of GasProcessingSubSystem Types

$gL = 1, \dots, GL$: set of future OilProcessing Locations

$w = 1, \dots, W$: set of future WaterProcessing Locations

$p = 1, \dots, P$: set of future GasTakeaway Locations

$s = 1, \dots, S$: set of future OilTakeaway Locations

Parameters

A: Annual production target

gor: projectd future WellProductionSubsystem GOR

wc: projectd future WellProductionSubsystem watercut

m_z: future WellProductionSubSystem estimated initial rate of production (Barrels per Day)

P: Aggregated current oil production of current WellProduction objects,

G: Aggregated current gas production of current WellProduction objects, calculated using P and GOR object attributes

W: Aggregated current water production of current WellProduction objects, calaculated using P and watercut object attributes

MP : Aggregated capacity of currently installed OilProcessingSubSystem objects

MW: Aggregated capacity of currently installed WaterProcessingSubSystem objects

MG : Aggregated capacity of currently installed GasProcessingSubSystem objects

MT: Aggregated capacity of currently installed OilTakeawaySubSystem objects

MP: Aggregated capacity of currently installed GasTakeawaySubSystem objects

XB_b: set of future OilProcessingSubSystem Types

VB_b: set of future OilProcessingSubSystem Capacities

DB_s: set of future OilProcessingSubSystem Capital Costs

XG_g: set of future GasProcessingSubSystem Types

VG_g: set of future GasProcessingSubSystem Capacities

DG_g: set of future GasProcessingSubSystem Capital Costs

VW_w: set of future WaterProcessingSubSystem Capacities

DW_w: set of future WaterProcessingSubSystem Capital Costs

VS_s: set of future OilTakeawaySubSystem Capacities

DS_s: set of future OilTakeawaySubSystem Capital Costs

VP_p: set of future GasTakeawaySubSystem Capacities

DP_p: set of future GasTakeawaySubSystem Capital Costs

Decision variables

$d_z = \begin{cases} 1 \\ 0 \end{cases}$ Drill WellProductionSubSystem location or not

$ib_{bTbL} = \begin{cases} 1 \\ 0 \end{cases}$ Install OilProcessingSubSystem type at location or not

$ig_{gTgL} = \begin{cases} 1 \\ 0 \end{cases}$ Install GasProcessingSubSystem type at location or not

$iw_w = \begin{cases} 1 \\ 0 \end{cases}$ Install WaterProcessingSubSystem location or not

$u_s = \begin{cases} 1 \\ 0 \end{cases}$ Execute OilTakeawaySubSystem contract for location or not

$e_p = \begin{cases} 1 \\ 0 \end{cases}$ Execute GasTakeawaySubSystem contract for location or not

Objective Function

Minimize capital spent on facility subsystems

$$\sum_{p=1}^{P} e_p * DP_p + \sum_{w=1}^{W} i_w * DW_w + \sum_{gL}^{GL} \sum_{gT=1}^{GT} ig_{gTgL} * DG_g + \sum_{bL}^{BL} \sum_{bT=1}^{BT} ib_{bTbL} * DB_b$$

Subject to:

$$\sum_{z=1}^{Z} b_z * m_z + P \leq A$$

$$\sum_{z=1}^{Z} b_z * m_z + P \leq MP + \sum_{bL}^{BL} \sum_{bT=1}^{BT} ib_{bTbL} * VB_b$$

$$(\sum_{z=1}^{Z} b_z * m_z) * w + W \leq MW + \sum_{w=1}^{W} i_w * VW_w$$

$$(\sum_{z=1}^{Z} b_z * m_z) * gor + G \leq MG + \sum_{gL}^{GL} \sum_{gT=1}^{GT} ig_{gTgL} * DG_g$$

$$\sum_{z=1}^{Z} b_z * m_z + P \leq MP + \sum_{s=1}^{S} u_s * VS_s$$

$$(\sum_{z=1}^{Z} b_z * m_z) * gor + G \leq MT + \sum_{p=1}^{P} e_p * DP_p$$

`systemEvaluate.py`

After the decision optimization program executes through all project years, the annual systems are enumerated through using an evaluation function. This function calls several system class methods which return sums or calculation as attributes for the system at that point in time.

```
def evaluateAnnualInstance(annualSystemInstance,seed = None):
    annualSystemInstance.calculateProduction()
    annualSystemInstance.calculateRevenue()
    annualSystemInstance.evaluateCapexDrilling()
    annualSystemInstance.evaluateCapexFacilities()
    annualSystemInstance.calculateEnergyRequired()
    annualSystemInstance.evaluateEnergyCreated()
    annualSystemInstance.calculateFuelCostsAndEmissions()
```

Appendix B: Multi-Integer Optimization Program for Expanded Model

Production, Oil and Compression Only - Type and Location Optimization Framework

Sets

$z = 1, \dots, Z$: set of Well Pad Locations

$l = 1, \dots, L$: set of Potential Component Locations

$t = 1, \dots, T$: set of Tank Battery Types

$s = 1, \dots, S$: set of Compressor Station Types

Parameters

m_z : Well Pad rate of production (Barrels per Day)

dzt_{zl}: Euclidean distance between Well Pad Location and Locations

ct_t: Cost of Tank Battery Types

Mt_t : Capacity of Tank Battery type

GOR: Gas to oil ratio

dts_{ll}: Euclidean distance between Tank Battery Location and Compressor Station

cs_s: Cost of Compressor Station Type

Ms_s : Capacity of Compressor Station type

f: Cost per foot of pipeline ($/foot)

Decision variables

xzt_{zl} : Production rate from Well Pad Location to Tank Battery Location

xts_{ll} : Gas rate from Tank Battery Location to Compressor Station Location

$bt_{lt} = \begin{cases} 1 \\ 0 \end{cases}$ Tank Battery of type is built at Location or not

$bs_{ls} = \begin{cases} 1 \\ 0 \end{cases}$ Compressor Station of type is built at Location or not

Objective Function

Minimize CAPEX cost

$$\sum_{l=1}^{L}\sum_{t=1}^{T} ct_t * bt_{lt} + \sum_{z=1}^{Z}\sum_{l=1}^{L}\sum_{t=1}^{T} bt_{lt} * dzt_{zl} * f + \sum_{l=1}^{L}\sum_{s=1}^{S} cs_s * bs_{ls} + \sum_{l=1}^{L}\sum_{l=1}^{L}\sum_{s=1}^{S} bs_{ls} * dst_{ll} * f$$

Subject to:

$$\sum_{l=1}^{L} x\_zt_{zl} = m_z, \qquad \forall z = 1, \dots, Z$$

$$\sum_{l=1}^{L} x\_ts_{ll} = \sum_{z=1}^{Z} x\_zt_{zl} * GOR, \qquad \forall l = 1, \dots, L$$

All production must be processed: For each well pad the production produced there must be equal to the production that arrives at the tank batteries. For each tank battery the gas separated there must be equal to the gas that arrives at the compressor stations.

$$\sum_{z=1}^{Z} x\_zt_{zl} \leq \sum_{t=1}^{T} M_{t_t} * b_{t_{lt}} \qquad \forall l = 1, \dots, L$$

$$x\_zt_{zl} \geq 0, \qquad \forall z = 1, \dots, Z, \quad \forall t = 1, \dots, L$$

$$\sum_{l=1}^{L} x\_ts_{ll} \leq \sum_{s=1}^{S} M_{s_s} * b_{s_{ls}} \quad \forall l = 1, \dots, L$$

$$x\_ts_{ll} \geq 0, \qquad \forall l = 1, \dots, L, \quad \forall l = 1, \dots, L$$

Interface & Capacity Constraints of Locations: The production rate to each tank battery location must be less than the capacity of that tank battery type. If it is not built it must be zero and cannot be negative. The gas rate to each compressor station location must be less than the capacity of the compressor station type. If it is not built, the gas rate must be zero and cannot be negative.

$$b\_t_{lt} \in \{0,1\}, \quad \forall t = 1,2,3, \quad \forall l = 1, \dots, L$$

$$b\_s_{ls} \in \{0,1\}, \quad \forall s = 1,2,3, \quad \forall l = 1, \dots, L$$

Binary Build Decisions: Build decisions must be binary for all tank battery locations and types

$$\sum_{1}^{3} b_{t_{lt}} + \sum_{1}^{4} b_{s_{ls}} \leq 1, \qquad \forall l = 1, \dots L$$

Co-Location Constraints: Tank Batteries and Compressor Stations cannot occupy the same location and only one type of component can occupy that location.

References

Crawley, E., Cameron, B., & Selva, D. (2016). *System Architecture: Strategy and Product Development for Complex Systems* (illustrate). Pearson, 2016.

de Neufville, R., & Scholtes, S. (2011). *Flexibility in engineering design*. MIT Press.

Ernst & Young. (2014). *Spotlight on oil and gas megaprojects Oil and gas capital projects series*. 16. http://www.ey.com/Publication/vwLUAssets/EY-spotlight-on-oil-and-gas-megaprojects/$FILE/EY-spotlight-on-oil-and-gas-megaprojects.pdf

Ghahremani, M., Aghaie, A., & Abedzadeh, M. (2012). Capital Budgeting Technique Selection through Four Decades: With a Great Focus on Real Option. *International Journal of Business and Management*, *7*(17).

Global Gas Flaring Reduction Partnership. (2019). *GGFR Technology Overview – Utilization of Small-Scale Associated Gas*. http://documents.worldbank.org/curated/en/469561534950044964/pdf/GGFR-Technology-Overview-Utilization-of-Small-Scale-Associated-Gas.pdf

Grobshtein, Y., Perelman, V., Safra, E., & Dori, D. (2007). Systems modeling languages: OPM versus SysML. *2007 International Conference on Systems Engineering and Modeling, ICSEM '07*, 102–109.

Hubbard, R. (2009). The role of gas processing in the natural-gas value chain. *JPT, Journal of Petroleum Technology*, *61*(8), 65–71. https://doi.org/10.2118/118535-JPT

International Council of Systems Engineering (INCOSE). (n.d.). *Systems Engineering*. Retrieved June 23, 2020, from https://www.incose.org/systems-engineering

Ipsmiller, E., Brouthers, K. D., & Dikova, D. (2019). 25 Years of Real Option Empirical Research in Management. *European Management Review*, *16*(1), 55–68. https://doi.org/10.1111/emre.12324

Myers, S. C. (1977). Determinants of corporate borrowing. *Journal of Financial Economics*, *5*(2), 147–175. https://www.sciencedirect.com/science/article/pii/0304405X77900150

Racheva, Z., & Daneva, M. (2010). How do real options concepts fit in agile requirements engineering? *8th ACIS International Conference on Software Engineering Research, Management and Applications, SERA 2010*, 231–238. https://doi.org/10.1109/SERA.2010.37

Savage, S. L. (2009). *The flaw of averages : why we underestimate risk in the face of uncertainty*. Wiley.

Stern, D. I., & Kander, A. (2012). The role of energy in the industrial revolution and modern economic growth. *Energy Journal*, *33*(3), 125–152. https://doi.org/10.5547/01956574.33.3.5

Tan, J., Otto, K., & Wood, K. (2017). A comparison of design decisions made early and late in development. *Proceedings of the International Conference on Engineering Design*, *2*(DS87-2), 41–50.

United States Environmental Protection Agency. (2009). Mandatory reporting of greenhouse gases. *Federal Register*, *74*(209), 56260–56519.

United States Environmental Protection Agency. (2018). *Emission Factor for Greenhouse Gas Inventories*. https://www.epa.gov/sites/production/files/2018-03/documents/emission-factors_mar_2018_0.pdf

World Bank Commodities Price Forecast. (2020). World Bank. http://pubdocs.worldbank.org/en/633541587395091108/CMO-April-2020-Forecasts.pdf

World Energy Outlook 2019. (2019). *International Energy Agency*. https://www.iea.org/reports/world-energy-outlook-2019

Wrigley, E. A. (Edward A. (2010). *Energy and the English Industrial Revolution*. Cambridge University Press.