NTNU
Norwegian University of
Science and Technology

An open-source approach to Integrated Operations

Brage Strand Kristoffersen

Petroleum Geoscience and Engineering

Submission date:	June 2017
Supervisor:	Vidar Hepsø, IGP
Co-supervisor:	Jon Kleppe, IGP

Norwegian University of Science and Technology
Department of Geoscience and Petroleum

Preface

This is a Masters thesis which will result in a Masters degree in Reservoir Technology and Petrophysics at the Norwegian University of Science and Technology (NTNU). The work was carried out in the spring semester of 2017. The topic of research was selected at NTNU with inspiration drawn from the on-going project of the Open Porous Media (OPM) Initiative.

The work is intended for everyone with an interest for Integrated Operations, especially the software development part. Basic knowledge of reservoir simulation and associated parameters are recommended, but not required to understand the fundamental ideas presented in this paper.

Trondheim, 02-06-2017

Brage Strand Kristoffersen

Acknowledgment

First and foremost I would like to thank my supervisor Professor II Vidar Hepsø as well as co-supervisor Professor Jon Kleppe. The technical input and the people I have got to know through them have been invaluable for this thesis. Without them, I would never have been able to complete such an enormous endeavour. Vidar has given valuable insight into the topic of Integrated Operations, and has been incredible in terms of discussing and giving me the freedom to learn and solve problems.

Further, I would like to thank Alf Birger Rustad at Statoil and Open Porous Media Initiative for countless meetings and continued feedback. I appreciate all the emails and giving a direction with my thesis from early on.

When in doubt, ask someone with more experience. A big thanks go to both post-doctoral researcher Mathias Bellout and PhD student Einar Baumann. Mathias showed great enthusiasm and helped me with the structure of the thesis. I am also incredibly grateful for all the technical insight and discussions I had with Einar!

Thank you also goes out to the interviewees at Statoil and AkerBP for their willingness to contribute and their open attitude while answering in-depth questions.

A huge appreciation goes to my girlfriend Silje Marie Tinderholt for keeping me motivated and always looking at the bright side of life.

B.S.K.

(Brage Strand Kristoffersen)

Summary and Conclusions

This thesis is based on exploring the topic of Integrated Operations (IO) and software development in the oil and gas industry. The philosophy of IO and how it impacted the oil and gas industry in the last decade is investigated. The term was introduced in the early 2000s, and builds on an old idea; allow engineers to collaborate in cross-disciplinary teams, and enable them to access all data, independent of their geographical location. IO is a philosophy to solve the problems of tomorrow concerning the governance, work processes, technology, and people. The investigation show that engineers and data have become increasingly intertwined as rising amount of data have become available, and the communication between offshore and on-shore facilities have significantly improved.

Through the last decade, companies of all industries have boosted their attention on information acquisition and data management. This increase in focus for information technology is often explained with the term digitalization. The idea of digitalization and Integrated Operations share multiple aspects; IO could be interpreted as a holistic approach which includes digitalization as the enabling dimension.

The implementation of Integrated Operations has seen a number successes and a comparable number of failures. Challenges associated with the deployment of IO related projects are often caused by the lack of adequate technological solutions. Innovation is the driving force behind change; it improves efficiency and increases the abilities of engineers. In many domains of the oil and gas industry innovation frequently happen, in other areas they do not. Reservoir engineering is an example of the latter. Typical reservoir engineering workflows involve simulating fluid behaviour in a reservoir and Eclipse is a tool. The simulator has been a workhorse of the industry for more than two decades, and have enabled large-scale field development and an increase in value realization.

Schlumberger uses a proprietary model to distribute Eclipse, a secure system to manage, but might also hamper innovation. This thesis treats the topics of software development models and distribution licenses. An alternative approach to the proprietary development model is to use an open-source model, which is much harder to sustain but stimulates innovation. The Open Porous Media (OPM) Initiative is currently developing an alternative open-source reservoir simulator named Flow. Flow offers unique opportunities and may offer Eclipse competition in the future.

In a benchmark study conducted on the Norne full-scale reservoir model, the results show that the open-source alternative, Flow, delivers both accurate and consistent results compared to the reference simulator, Eclipse. The performance

shows that in a single-core environment, Eclipse slightly outperforms Flow concerning speed, while the opposite applies to a dual-core environment. These results give an indication of potential that may materialise when an open-source framework and mindset is applied to a domain previously dominated by a proprietary software solution.

The structure of open-source software development increases the pace at which features and the core capabilities are developed. Open-source software is rarely built from scratch, rather, it is most commonly built on other projects with the same development model. This structure creates co-dependencies between multiple projects and allows the scope of one project to be maintained at the core functionality. Flow applies a modular toolbox for solving partial differential equations (PDEs) with grid-based methods, called DUNE (DUNE, 2017). DUNE is an independent project that continuously evolves and optimizes its capabilities, which through a co-dependency contributes to a better framework for Flow and improved run-times for simulations.

Among the biggest challenges in an open-source development model is to establish a sustainable business model and create a reputation as a reliable alternative. In a project like OPM, it is vital to find partners to expand, develop and verify. This involves a great deal of risk for OPM, but not necessarily for those who would partake in such an endeavour. OPM offers a unique opportunity for oil and gas companies to secure their ability to operate. Giving greater leverage towards existing suppliers and avoid a vendor lock-in situation.

As of today, OPM does not deliver all the tools that are needed for a field development. It does, however, give an insight into what the future may bring regarding software. An open-source alternative allows for full transparency regarding how the software operates and thereby could increase the trust by the engineers that utilize the software. This increased trust could in the future increase the amount of automation in typical workflows and transition engineers from doing iterative tasks to value adding activities.

A fundamental part of the open-source software is its openness towards modifications. This would allow an operator to modify the simulator to fit the needs of individual assets. Also, support for custom scripts and third-party programs could have native support and deliver additional information and increase the functionality.

Sammendrag og konklusjon

Denne Masteroppgaven utforsker teamet Integrerte Operasjoner (IO) med et spesielt fokus på programvareutvikling innenfor oljeindustrien. En grundig gjennomgang av ideen bak filosofien og hvordan den har påvirket oljeindustrien er foretatt. Begrepet Integrerte Operasjoner ble innført tidlig på 2000-tallet. Ideen bak begrepet er gammel, den baserer seg på at ingeniører skal kunne jobbe i multidisiplinære team og ha tilgang til data uavhengig av geografisk posisjon. Filosofien er ment å løse morgendagens problemer i form av organisasjonsstruktur, folk, arbeidsprosesser og teknologi. Gjennom undersøkelsen foretatt i denne masteroppgaven viser det seg at forbedret kommunikasjon mellom offshore og landbaserte anlegg har ført til økt tilgang på informasjon, og tilrettelagt for bedre samhandling mellom ingeniører og data.

Gjennom det siste tiåret har selskaper i alle industrier økt sitt fokus på informasjonsinnsamling og håndtering av data. Denne økningen forklares ofte med begrepet digitalisering. Ideen bak digitalisering og Integrerte Operasjoner deler mange aspekter, og en kan se på IO som en helhetlig tilnærming som inkluderer digitalisering som den muliggjørende dimensjonen.

Implementeringen av Integrerte Operasjoner har sett stort antall suksesser og et nesten like stort antall mislykkede initiativer. Utfordringene i implementeringen av disse initiativene skyldes ofte utilstrekkelige teknologiske løsninger. Innovasjon er drivkraften bak endring; det øker effektiviteten og evnene til ingeniører. I noen fagfelt innenfor oljeindustrien skjer innovasjon med stor frekvens, i andre skjer det sjeldent. Reservoar simulering er et eksempel på det siste. De vanligste arbeidsprosessene innenfor dette domenet involverer å simulere fluiders oppførsel i et hydrokarbon reservoar. Eclipse er et verktøy som benyttes til dette, og har vært en arbeidshest for industrien mer enn to tiår. Eclipse har lagt fundamentet for storskala utbygninger og gjort det mulig å realisere mer verdi enn tidligere.

Schlumberger bruker en proprietær modell for å distribuere Eclipse, det gir enkel kontroll inntekter, samtidig som det reduserer muligheten for innovasjon innenfor fagfeltet. Denne masteroppgaven undersøker forskjellige programvareutviklingsmodeller og lisenser. Et alternativ til den proprietære utviklingsmodellen er å benytte seg av en open-source programvareutviklingsmodell. Denne modellen har utfordringer i form av direkte inntekter, men stimulerer innovasjon. Flow, en alternativ open-source reservoarsimulator, utvikles for øyeblikket av The Open Porous Media (OPM) Initiative. Flow tilbyr unike muligheter for operatører og vil i fremtiden være en direkte konkurrent til det nåværende monopolet.

I en verifikasjonsstudie gjennomført på Norne full-felt reservoarmodell viser resultatene at open-source alternativet, Flow, er i stand til å levere både nøyaktige

og pålitelige resultater sammenlignet med referansesimulatoren, Eclipse. Ved kjøring på en kjerne er Eclipse litt raskere enn Flow. Ved kjøring på to kjerner er resultatene motsatt, her viser Flow bedre ytelse enn Eclipse. Disse resultatene gir en indikasjon for potensialet som kan materialiseres om et open-source rammeverk og innstilling benyttes på et domene som tidligere har vært dominert av proprietær programvare.

Open-source programvareutvikling har en struktur som øker takten på utviklingen av både kjerneegenskaper og ny funksjonalitet. Open-source programvare utvikles sjeldent fra bunnen, det er vanlig å basere utviklingen på moduler fra andre prosjekter med samme utviklingsmodell. Denne strukturen skaper koblinger mellom flere prosjekter og tillater et prosjekt å fokusere på kjernefunksjonaliteten. Flow benytter seg av et modulært verktøy kalt DUNE; for å løse partielle differensialligninger (PDEs) med grid-baserte metoder (DUNE, 2017). DUNE er et uavhengig prosjekt som kontinuerlig utvikler seg, dette fører det til at Flow får et forbedret rammeverk som igjen øker simulatorens ytelse.

Blant de største utfordringene i et open-source programvareutviklingsprosjekt er å få i stand en bærekraftig forretningsmodell og skape et rykte som et pålitelig alternativ. I et prosjekt som OPM, er det viktig å finne partnere som kan hjelpe med utvidelse, utvikling og verifisering. Dette involverer en stor grad av risiko for OPM, men ikke nødvendigvis for partnerne som er villig til å bli med. OPM tilbyr en unik mulighet for å sikre evnen til å operere. Dette gir en bedre forhandlingsposisjon overfor eksisterende Service selskaper og hjelper operatører med å unngå lock-in situasjoner.

I dag leverer ikke OPM alle de verktøyene som trengs for å gjennomføre en feltutbygging. Prosjektet gir derimot verdifull innsikt i hva fremtiden kan bringe i form av programvare. Et open-source alternativ gir full innsikt i hvordan programvaren fungerer, og kan dermed oppnå økt tillitt blant ingeniører. Dette vil på lang sikt kunne føre til ytterligere automatisering av arbeidsprosesser, og kan endre rollen til en ingeniør fra å gjøre iterative oppgaver til å øke fokuset på verdiskapende aktivitet.

En viktig del av open-source programvare er muligheten til å tilpasse programvaren. Dette gjør operatører i stand til å justere simulatorene etter behovene til individuelle reservoarer. I tillegg, vil støtte for tredjeparts programvare og egenutviklede skript føre til at simulatorer kan gi ytterligere informasjon og øke funksjonaliteten.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

During the last decade, the petroleum industry has undergone significant changes. Both at a project level and an organizational level, substantial resources have been committed to increasing the level of collaboration and cross-disciplinary communication in any large-scale endeavors. This change was primarily influenced by the introduction of the term Integrated Operations. Integrated Operations is a movement that seeks to eliminate disciplinary silos and instead increase awareness between all disciplines of engineering. This is to ensure improved decision-making, and also to create an effective collaborative team. Thanks to fiber optic telecommunication, all available data are accessible regardless of their geographical location.

Integrated Operations have had a lot of success-stories but the implementation has not been as predictable as first thought. Limitations within technology and software are challenges that must be addressed to embrace the philosophy fully.

Through the thesis, the topic of Integrated Operations is explored. A chapter is dedicated to giving an introduction to fundamental principles surrounding the philosophy and the dimensions in which it focuses. It will review some of the challenges associated with the deployment and further investigate whether one of the keys to increased innovation could be found in software development. By focusing on software development and looking at relevant examples from other industries, it will try to highlight some of the possibilities for innovation that could happen in a fundamental shift of software development model. From a proprietary model to an open-source model. One such open-source project is the Open Porous Media (OPM) Initiative. The thesis will review what OPM is and how it compares to the industry-leading reservoir software suites.

OPM is an umbrella of software, which contains several tools. this thesis, results from the reservoir simulator, Flow, will be reviewed by using the reservoir simulation tool, ResInsight. A benchmark study is conducted and presents a thorough review of the performance and validity of the reservoir simulator. This will be done using the challenging Norne full-field reservoir model comparing it with the current benchmark simulator, Eclipse.

This thesis will try to answer whether one of the keys for the accelerated deployment of Integrated Operations is located in the development model of the software, by changing the principle of the simulator from a black-box approach to a transparent white-box approach. This would allow for for modifications that in the long-run could increase trust and renew innovation within the domain of reservoir engineering.

1.2 Objectives

The main objectives of this Master's project are:

1. Explore the topic of Integrated Operations and analyze the challenges that the upstream industry is experiencing.
2. Conduct interviews to get impressions on how the industry itself views Integrated Operations, open-source software and the willingness to contribute towards more openness in the market of reservoir simulation.
3. Benchmark Flow and compare the results to the current reference reservoir simulator, Eclipse. Study both the validity and performance of the simulator.
4. Investigate whether a change in development model, from proprietary to open-source, can increase confidence and trust towards the simulated results and in the long-term increase automation in order to transition reservoir engineers from doing iterative tasks to decision-making.
5. Study how the introduction of an open-source competitor can alter the dynamics of the current market for reservoir simulation.

6. Investigate the limitations and potential business models that are associated with an open-source development model.

1.3 Limitations

The limitation of this study involves lack of observations at different companies and the fact that several major operators, vendors, and service providers on the Norwegian Continental Shelf (NCS) posesses several internal technologies, processes, and software. These therefore can not be studied in the same way as none-internal alternatives.

As the author is only experienced within the NTNU educational system and to a limited degree Statoil, Memorial University of Newfoundland and Labrador. This lack of insight into other oil provinces limits the area of impact, and will, therefore, not be representative of the industry outside the NCS. However, many of the concepts described here are generic; there could be the analogous use of this for other oil and gas provinces.

1.4 Method

There are five methods of empirical data that supports this project work:

First, the author has educational experience within reservoir engineering and general courses in petroleum engineering, which allows for an understanding of the industry and awareness of the needs of multiple branches of subsurface engineering.

Second, there is an extensive literature review, which analyzes the requirements and effects of Integrated Operations (Grimstad, Bjarne, et al. 2014; Henderson, John, Vidar Hepsø, and Øyvind Mydland, 2013). These have been studied at length and further supports the conclusions and discussion.

Third, for this project, there have been talks with experienced professionals and project leaders at Statoil ASA. These discussions have mainly focused on the open-source development of the Open Porous Media (OPM) Initiative, Integrated Operations at Statoil and challenges that professionals face on a day-to-day basis. The combination of these empirical methods makes for a solid fundament further discussion and conclusions.

Fourth, reservoir simulations from Eclipse and Flow have been conducted and the output analyzed to determine viability of the open-source reservoir simulator, Flow.

Fifth, interviews have been conducted with four experienced professionals and researchers at Statoil ASA and AkerBP ASA.

1.5 Structure of the Thesis

Chapter 1 "Introduction": Contains the background, objectives, limitations, methods and structure of the thesis.

Chapter 2 The context of Integrated Operations: This chapter explain some of the key concepts of Integrated Operations (IO), what differentiates this philosophy from traditional practice and what challenges are faced when trying to adapt to new, increasingly automated, workflows. (This section is an improved version of former work in the specialization project, Kristoffersen (2016))

Chapter 3 Introduction to Software Development: This chapter introduces the practice of software development in two fundamentally different models. It highlights the benefits and drawbacks of open-source and proprietary development models. The chapter also reviews some common licenses and utilizes examples from other industries to highlight potential business models for an open-source competitor. (This section is an improved version of former work in the specialization project, Kristoffersen (2016))

Chapter 4 Norne Benchmark Study: Contains a case study of an open-source and a proprietary reservoir simulator, namely Flow and Eclipse. Eclipse is the current reference simulator and is therefore used as a basis to compare results of the same simulation with Flow. This chapter uses a statistical approach, relative deviation, and cell-by-cell comparison to verify the solution. The run-time is also considered. The structure of the chapter is like a report, with introduction, method, results and a small discussion section.

Chapter 5 Discussion: This chapter contains the main discussion of the thesis. It brings all the elements from the above chapter together to evaluate the current situation and discuss the possibilities that is brought to the table by making open-source competitors a real alternative, and what the future deployment of Integrated Operations may bring. (One paragraph on security is based on former work in the specialization project, Kristoffersen (2016))

Chapter 6 Summary and conclusion: Contains the end-summary and concluding remarks. It also gives suggestions on further work and where to find the raw output from the simulations.

Appendix A Acronyms and abbreviations: Appendix containing the most commonly used acronyms and abbreviations.

Appendix B Formation description: Contains information about the formations found in the Norne hydrocarbon reservoir.

Appendix C "Ensemble": Contains the values that were altered from the base case. Both original and altered values are included.

Appendix D "Production data and pressure profile: Contains the raw data extracted from both simulators.

Appendix E "Saturation and Static Properties": Contains an overview over the saturation of oil, gas, and water. In addition, it contains an overview over the difference in static properties, namely the transmissibility.

Appendix F "References": Contains all the references used in support of the thesis.

Chapter 2

The context of Integrated Operations

Through this segment, there will be a focus on what Integrated Operations (IO) involves, and on the relevant dimensions of an organization of which it operates. The chapter will also highlight the challenges of deploying IO, and investigate whether the deployment has stagnated in between generations.

2.1 Introduction to Integrated Operations

Traditionally, oil and gas companies have organized different disciplines and functions into silos, working in serial to complete projects or similar complex endeavors. These silos were free to operate as they pleased; within their domain. However, communication between silos was going only through management and formal channels, making the process of acquiring input from other relevant engineering disciplines a slow and tiresome process. Goals were set locally, working only towards Key Performance Indicators (KPI) established by the silo itself. These KPIs facilitated an environment where input and output from other functions were assumed, limiting the collaborative effort.

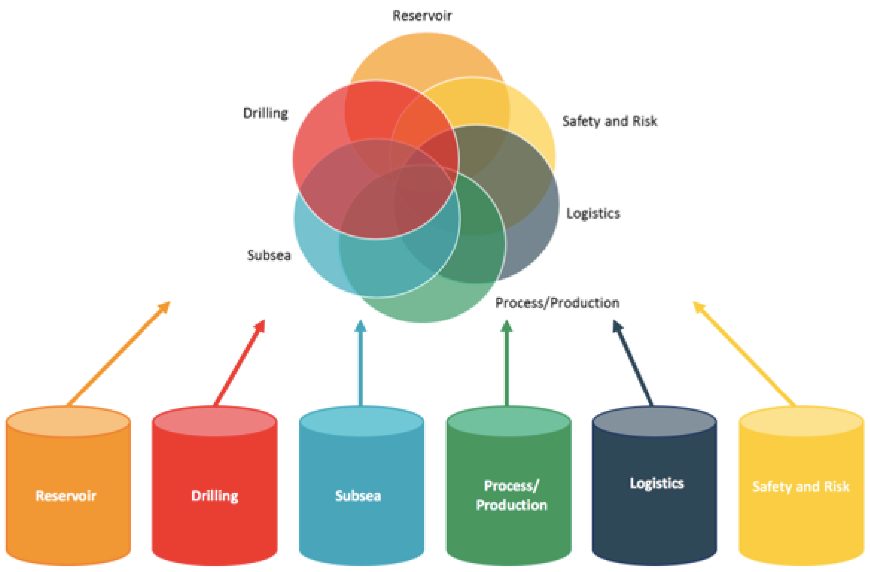


Figure 2.1: The end-goal of Integrated Operations - reducing work in functional silos and enhancing cross-disciplinary collaboration.

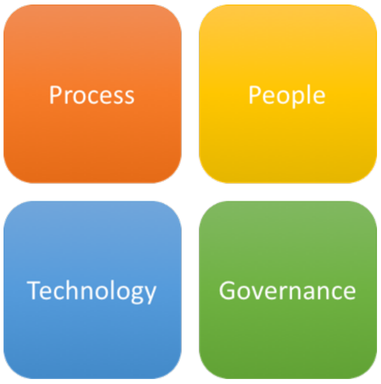


Figure 2.2: Dimensions of IO

Integrated Operations (IO) is a new design philosophy that seeks to eliminate these silos, and instead, make every discipline an integral part of every decision (Figure 2.1). Reducing the need for a serial workflow, and rather increase project efficiency by operating in parallel. It allows for fast-tracking of projects, enabling engineers to work on different critical activities simultaneously.

The notion of Integrated Operation was established to create a competitive advantage through clever use and adaption of existing organizations. IO processes facilitate communication, digitalization, and the achievement of common goals for the field as an entity. (Henderson, Hepsø, Mydland, 2013) IO works in multiple dimensions to achieve efficiency (Figure 2.2). It is important to stress that no dimension creates value by themselves, rather, through the adoption of all dimensions it the potential to set up and extract value.

2.1.1 The Information Ecology

Integrated Operations is an information ecology, a set of multiple capabilities or niches that exist within the oil and gas industry. The concept of ecology is used to depict the dynamics of emerging situations associated with Integrated Operations and the upstream industry (Hepsø, 2013).

Three distinct factors facilitate the information ecology of Integrated Operations; the increasingly smarter infrastructure and the ability to rapidly transfer data to where it needs to be; the standardization of telecommunication and software/hardware platforms; and the development of communicative tools that enables seamless collaboration between different geographical locations. (Hepsø,

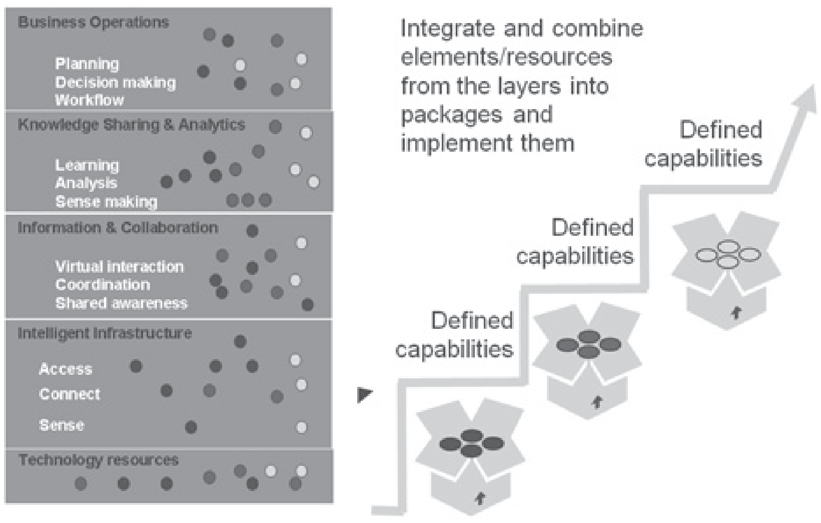
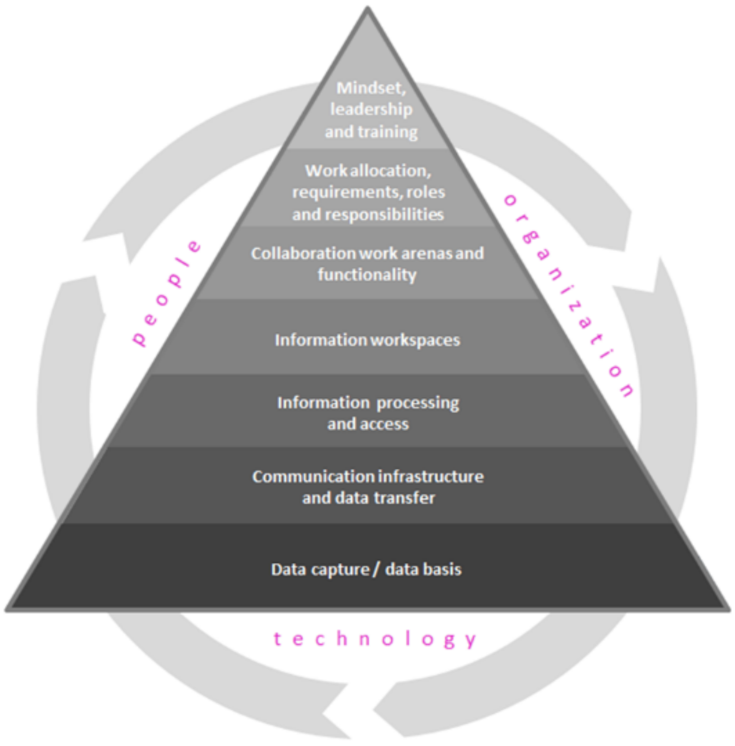


Figure 2.3: Generic stack-model as presented in Hepsø et al. (2013)

2016). Combining these three innovations allow for a move towards real-time data analysis and work flows.

Capability Platform

The capability platform is one way of relating the design of an organization to the mentioned ecology. A capability, as defined by Henderson, Hepsø, Mydland, (2013) is a set of interdependent activities involving people, process, technology, and and governance, which generate value through design efficiency; creates economic value through networks effects with other capabilities in the ecology; and has specific architectural control points that enable stakeholders to systematically capture portions of the economic value that has been created (Henderson, Hepsø, Mydland, 2013). Within the information ecology of Integrated Operations, there are many capabilities or niches; it is used to illustrate complex interactions inside the upstream industry.



Mindset, leadership and training
Work allocation, requirements, roles and responsibilities
Collaboration work arenas and functionality
Information workspaces
Information processing and access
Communication infrastructure and data transfer
Data capture / data basis
people
organization
technology

Figure 2.4: Success criterion for Integrated Operations developed by Statoil

The Layers

The capability stack can be divided into several basic layers, as shown in Figure 2.3; each niche dependent on the ones below. All layers represent a set of activities, which allows the leadership to convey attention to one particular layer. It can also provide vendors, operators, or service companies with business opportunities by creatively implementing all dimensions of IO to increase efficiency (Henderson, Hepsø, Mydland, 2013). The characteristics of a layer are highlighted in Table 2.1.

Table 2.1: Characterization of layers (Henderson, Hepsø, Mydland, 2013)

1.	Having a clear business proposition
2.	Clear interface to other layers, such that it can be decoupled
3.	The layer must present an active market for a solution
4.	Have a well-defined business metrics that reflects the core value proposition

There are seven success criteria developed for Integrated Operations (Figure 2.4), these must all be fulfilled to maximize the benefits from IO in each of the layers. These criteria also facilitates a good start for planning and executing large scale projects. In the planning phase, each of the layers have to be addressed. This would enable service companies and operators to develop a solution for one or more of the niches. By standardizing this, there could be tailor-made solutions or common packages found in several of ongoing projects. By offering standard packages that fulfills one of these layers, there would be more competition and operators would have greater competition between vendors.

Integrated Operations and Digitalization

Integrated Operations is in continues development, as new technology emerges and an increasing amount of data become available there is the need to manage it efficiently through the organization and in teams. Digitalization therefore is an integral part of IO. Without digitalization, IO would not be as efficient. IO, in contrast to digitalization, is an holistic approach, and it is not exclusive to the acquisition and management of data.. The philosophy of Integrated Operations is to combine technology, governance models, people, and processes in innovative ways to enable better management of assets.

2.1.2 People in Integrated Operations

In Integrated Operations, there are substantial benefits of having engineers understand the fundamentals of related disciplines of engineering (Carlile, 2004). By knowing what other function groups might need, it is possible to utilize and provide information with increased accuracy. There are significant amounts of information that should be conveyed through each domain. In Integrated Operations, this is well described with a T-model (Figure 2.5), the horizontal part of the T represents the broad knowledge about other engineering disciplines and tasks, this could be acquired through both formal and informal training. Informal training could be performed through anything from socialization to collaborative projects. The vertical part of the T represents the in-depth knowledge an individual has about their discipline of engineering. Both must be substantial enough to allow for efficient and non-ambiguous communication. If this is not the case, it will be difficult, for example, for a geophysicist to give accurate information about seismic to a reservoir engineer who is using this information for more accurate reservoir modeling.

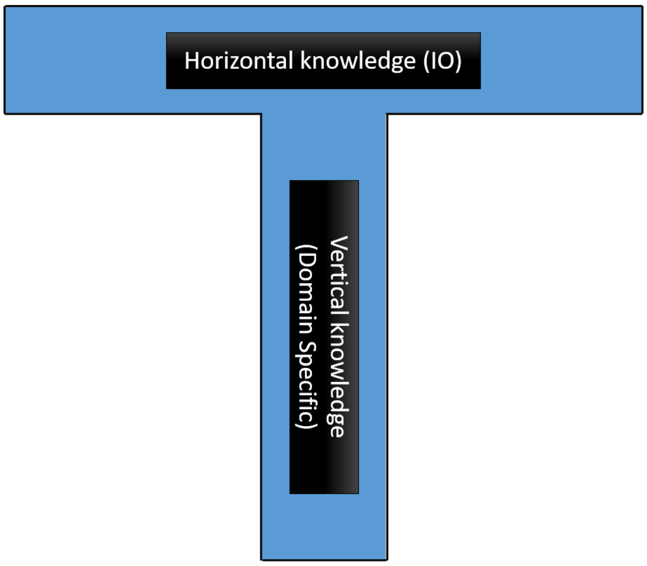

Horizontal knowledge (IO)
Vertical knowledge (Domain Specific)

Figure 2.5: T-model, a description of how engineers need to obtain both cross-domain knowledge and in-depth domain specific knowledge (Adept from Hepsø, 2016)

This example show how two very dependent disciplines might have a difficult time accurately understanding each other due to lack of cross-training and common knowledge.

An additional benefit of multidisciplinary training is that it makes it possible to work across engineering disciplines to solve problems that might not be apparent if it was being worked on by people from a single disciplines. As Dorothy Leonards (1995) stated, that most innovation happens at the boundaries between disciplines or specialization. Illustrated in Figure 2.7 is an example of a typical field development project. It shows the amount of information, and what extent of boundary knowledge individual disciplines need acquire, to effectively collaborate in multidisciplinary teams. One way of accommodating increased communication is to establish a framework. Paul R. Carlile (2004) described one such framework (Figure 2.6). The framework acknowledges that different domains have different lexicons, goals, and interests in terms of the task at hand. Creating complex products or services often require several different types of knowledge (Carlile, 2004). People have different interests as to what should be added and modified; e.g. flow assurance engineers would like a stable flow from wells to maintain stability in the riser, while the reservoir/production engineer would like to use well tests to increase their understanding of the subsurface environment. Both are important, although it is beneficial for both disciplines of engineering to have an equilibrium between these two actions. For this to happen, there must be knowledge transferred between boundaries.

Syntactic knowledge is information shared by a common lexicon, in which the meaning is rigid and non-ambiguous. This allows for the transfer of knowledge. However, as the novelty grows, and the lexicon is no longer sufficient to transfer knowledge, it transitions into the next type of boundary.

Semantic or interpretive Boundary occurs when novelty increases to the point where common lexicon is no longer sufficient to describe the outcome, and increased ambiguity makes the transfer of knowledge difficult. This boundary opens for interpretation by different disciplines, as different domains might have different meanings tied to different objectives. It is therefore important to create shared meanings such that negotiation is non-ambiguous, and an overview is created by all involved actors. Researchers have shown that by participating in similar activities, you develop shared meanings (Orr, 1996). This emphasis the need to work in across boundaries to solve issues, and thus creating a shared meaning, facilitating for innovation.

The pragmatic boundary arises when the increased novelty presents different interests (Carlile, 2004). If two different disciplines have different interests, e.g. Reservoir engineers and flow assurance engineers may have negative impact on

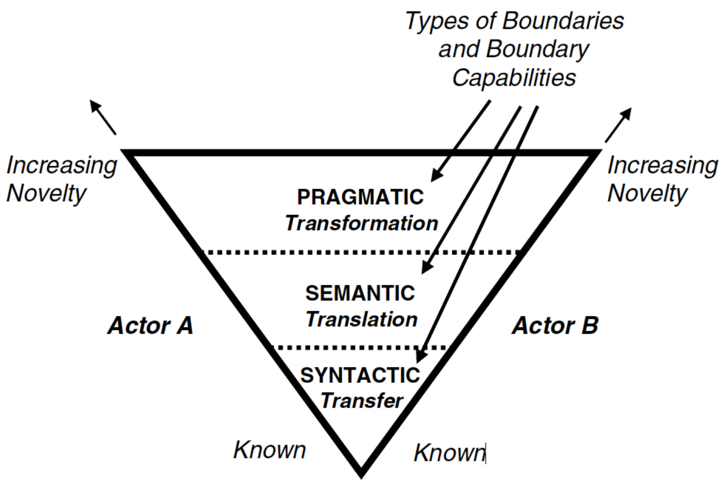


Figure 2.6: Framework for managing knowledge across boundaries (Carlile, 2004)

each other, since these two different disciplines have a conflicting interests. Interests of each domain are not clearly expressed to each other, due to lack of a common understanding. It might create a costly event, such as maintenance plans being rescheduled and not performed in a streamlined fashion.

These challenges may happen in a matrix structure, where the two types of leadership are present, one managing the economy, the other managing the technical solution. It causes a headache for engineers, therefore, by acknowledging and mapping all these boundaries, it is possible to identify problems and manage them between engineering domains, preventing conflicts. Problems must be identified before planning, and using collaborative tools to understand each others constraints and requirements. For Integrated Operations to unlock the potential efficiency, people must be convinced that new workflows contains improvement over the traditional workflows. Increasing the awareness of the capabilities and benefits of the new technology.

2.1.3 Change Management in Integrated Operations

Organizations and developments must be modeled after the fundamental success criterions in IO and can benefit greatly from a capability platform approach. These criterions should be in the mind-set of the management; in IO, this is called

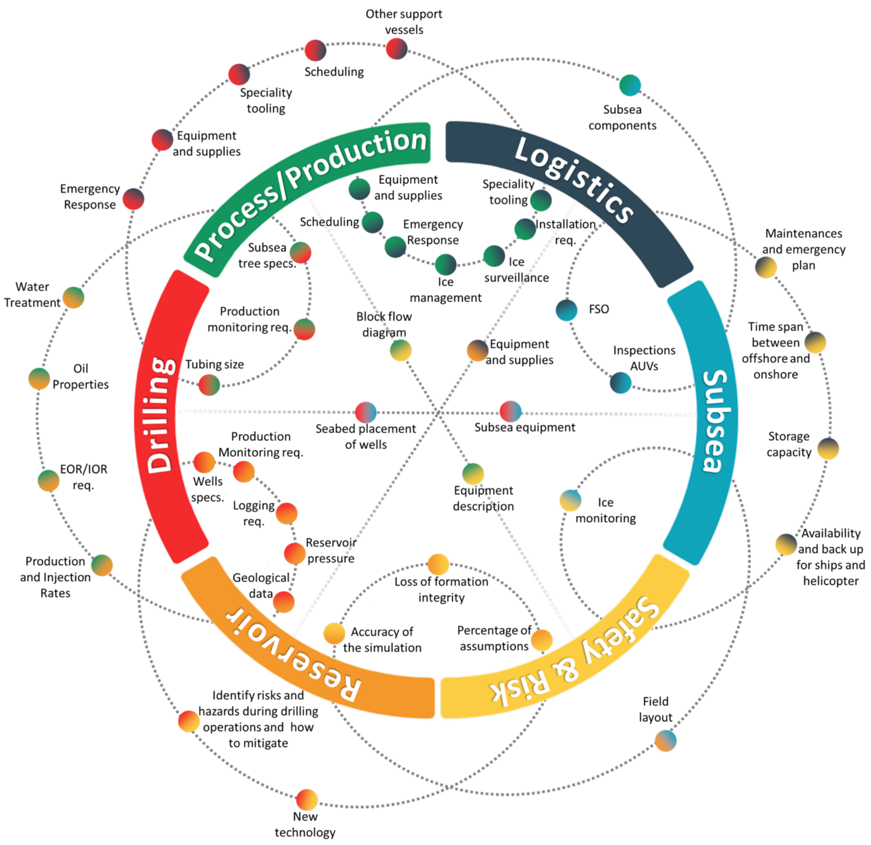


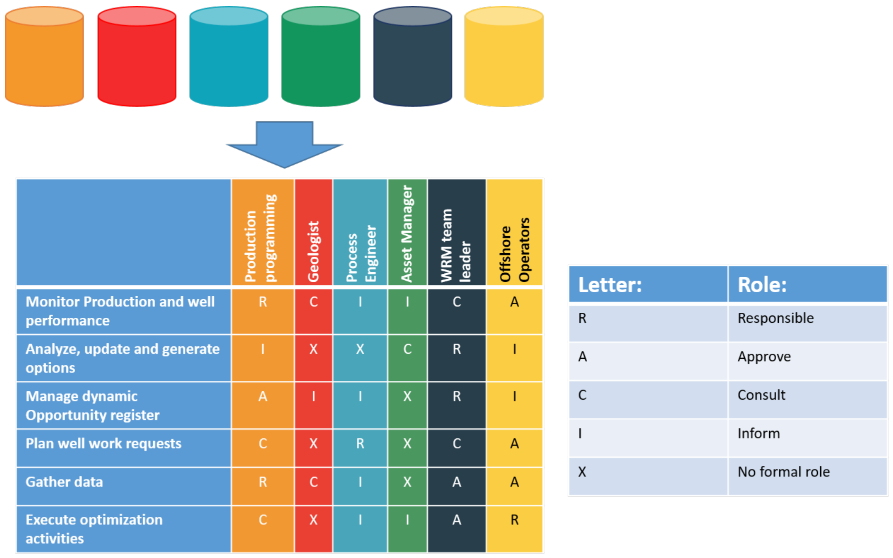
Figure 2.7: Boundary chart of the information that has to be conveyed in a typical field development project

change management. Change management is essentially moving an organization from the traditional way of working through a transition phase and adapting to work in the ways inspired by Integrated Operations. It is important to note that change management is a continuous process, to work and adjust every dimension, increasing efficiency. In turn, from a business point of view, this ability to effectively engage collaboration across boundaries without imposing a command and control process enables a significant increase in flexibility and innovation (Larsen, 2012). For this to happen, it requires a resolute organization and management that is aware of challenges that can occur during the transition phase. People are not inherently positive to change; one might, therefore, experience active or passive resistance even in the most adaptive organizations. Change management is a complicated process, as an organization moves through the transition phase, there must be a significant degree of commitment from employees. If people are simply doing as they are told, not because of their commitment to an organization, the contrary happens, and the potential efficiency increase decline with it. If, however, people are involved in the transition and adoption of new work processes and technologies, the commitment increases. (Rosendahl, Tom, et. al (2013)).

2.1.4 Governance in Integrated Operations

As video conferencing and collaborative work environments (CWE) are getting more common, and potential for cross-functional work is established there must be a redesign of the organizational structure (Guldemond, 2011). Guldemonds research showed that there is a common belief within the industry that the existing functional groups can still be the lines of communication, even in a CWE matrix structure (Figure 2.8). As CWEs are established, the engineers transition to working as a team, rather than as a function. The incentives and rewards should reflect this and reward cross-functional accomplishments, rather than appraising work done within the function (silo). This common belief is, therefore, inefficient; thus, creating new organizational structures may prove difficult. However, to realize the full potential of collaboration, it is necessary to redefine the organizational structure (Edwards, Mydland & Henriquez, 2010).

The deployment of Integrated Operations has also had a significant impact on management teams. The goal is to develop a shared situational awareness between on-shore and off-shore experts and management, through formal and informal contact in a collaborative work environment (e.g. continuous video conferencing and virtual collaboration rooms, smart boards, etc.). Through this effort, management and engineers' tasks get more of an overview of who knows what and can, therefore, work better as a team, not only as individuals.



	Production programming	Geologist	Process Engineer	Asset Manager	WRM team leader	Offshore Operators
Monitor Production and well performance	R	C	I	I	C	A
Analyze, update and generate options	I	X	X	C	R	I
Manage dynamic Opportunity register	A	I	I	X	R	I
Plan well work requests	C	X	R	X	C	A
Gather data	R	C	I	X	A	A
Execute optimization activities	C	X	I	I	A	R

Letter:	Role:
R	Responsible
A	Approve
C	Consult
I	Inform
X	No formal role

Figure 2.8: The move from functional to matrix structure with transparent responsibilities

In the philosophy of Integrated Operations teams should be smaller; have greater responsibilities and ownership of tasks. Kristin, an ongoing field development at Statoil, is an asset designed around the philosophy of Integrated Operations (Guldemond, 2011). This design involves having a high degree of ownership towards your responsibilities and tasks, as well as increasing the amount of transparency of competencies and tasks; in turn promoting a culture of knowledge sharing and boundary spanning. As the entire asset was designed from scratch with IO in mind, teams are smaller.

2.1.5 Technology in Integrated Operations

In Integrated Operations, the focus is to improve every dimension: process, people, technology, and governance. They all must be adapted to each other continuously. As mentioned in the introduction, technology alone does not create value. Through the smart implementation of technology to organization and work processes, value can be created and profited. Technology has played a central role in deploying workflows of Integrated Operations, as the enabling technology. Integrated Operations in tandem with digitalization; allows for closer cooperation between offshore and on-shore facilities. It also provides with the means of monitoring the behavior of an assent, since more data can be collected through an increasing number and more durable sensors.

2.1.6 Software in Integrated Operations

Software in Integrated Operations is a sub-category of technology; it plays an integral part for combining, communicating, and automating tasks of several subsurface disciplines. In its essence, it is what the engineer will use to complete workflows. It should ensure interoperability and easy-to-understand interfaces. In addition, this should be combined with the possibility of getting insight into what the software does. Today, there are multiple suites capable of doing this job, but as this these proposes, is that most software is black-boxed using proprietary licences. By black-boxing, engineers do not get to immerse themselves in the source code, and find themselves unable to adapt or modify the interface other than what the developer believes is necessary.

2.1.7 Deployment of Integrated Operations

Norwegian Oil industry Association (NOA) distinguished the different stages of implementation of Integrated Operations into three generations (NOA, 2005). Figure 2.9 illustrates the various generations of IO versus the realized value.

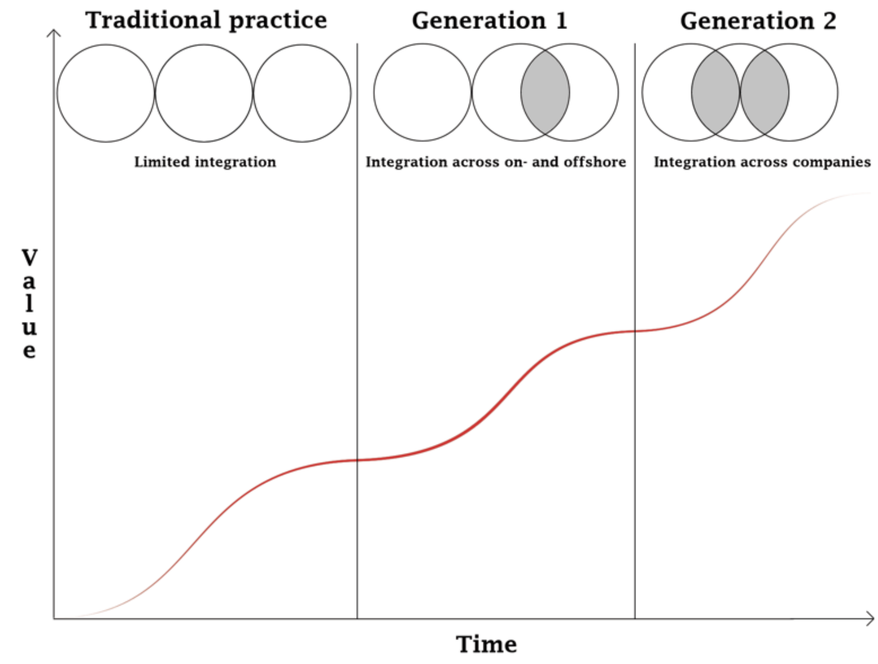


Figure 2.9: Illustration of the realization of value vs. generations in Integrated Operation (Adept from NOA, 2005)

Traditional practice: The oil and gas industry was, as mentioned above, divided into silos without the necessary communication between the different domains. This causes asynchronous development of assets.

Generation 1: This is the present generation. Innovations such as video conferencing, fiber optic cables, and multiphase flow meters (MFM) have made it possible to share more information between off- and onshore locations, allowing for greater cooperation and shared situational awareness. Increasing uptime of specialist centers beyond regular work hours and teams consisting of off- and onshore personnel that has mandated the necessary authority to make important decisions.

Generation 2: The next stage of deployment is the second generation of Integrated Operations. This generation seeks to integrate service companies, vendors, and operators in decision-making centers, increasing availability and sharing of information. Contracts have to be renegotiated; roles of suppliers and service companies must be redefined, and common data standards have to be developed. This will allow for service companies and suppliers to deliver better services than what has previously been possible. The focus should move on to automatization of primary tasks, for increased decision-making support, e.g. closed-loop reservoir management (Jansen, J. D., Brouwer, R., & Douma, S. G., 2009).

The classification of Integrated Operations was developed in 2005 by NOA, at that time it was suggested that Integrated Operations would reach the second generation of deployment within a couple of years and full implementation of automated processes during a decade. For several reasons this has not been the case because the upstream industry difficulties in automatization and decision support from software (Bjarne Grimstad, Petter Almklov, Bjarne Foss, Vidar Gunnerud, 2015). One of the fundamental assumptions in the 2005 report was the fact that people and organizations were considered the remaining factor. It was assumed that the technological solutions were already existent and that people and organizational structures had to adapt to utilize the potential of Integrated Operations. This assumption does not seem to reflect reality; experts have analyzed the situation and concluded that, as mentioned earlier, every dimension in the ecology must evolve simultaneously and in a continues manner (Hepsø, 2016). Technology on its own does not create value. Value is created through clever combination and implementation of all dimensions.

2.2 Challenges Associated with the Deployment of Integrated Operations

The Center for Integrated Operations has the mandate to identify problems in the implementation of Integrated Operations, comparing the upstream industry with the downstream and other industries (Grimstad, Almklov, Foss, Gunnerud, 2015). As the scope of this thesis is primarily focused on simulation and software, these challenges will be highlighted. In the following sub-chapter, a few of the common challenges will be addressed.

2.2.1 Resistance in the Organization

Resistance to change is a part of human behavior; people seek to maintain the status quo. (Buchanan & Huczynski 2010; Burns 2009; Cummings & Worley 2009). Kurt Lewin defined it as a restraining force moving in the direction of status quo (Lewin 1952, cited in; Piderit 2000, p.784) this might result in a lag of implementation. Rosendahl, Tom. Et al. (2013) argues that this might be a necessity by listening to employees that are reluctant to embrace new work processes and technology management can evaluate critical opinions and improve the final state.

2.2.2 Batch vs. Real-time Data Usage

The last decade the industry has seen a massive increase in information gathering. Greenfields now equipped with instrumentation for flow, temperature, and pressure monitoring. There are vast quantities of data acquired. This data, however, is being used in a sub-optimal way, relying on batch-wise history matching and optimization. This sub-optimal usage reduces the effectivenesss of the decisions that are to be made, compared to using data real-time and always having automatically history-matched models. By eliminating the manual history matching and instead replace this iterative workflow with automated computer processing. This automatization would enable reservoir engineers to work with decision-making rather than data matching. One such workflow for an asset is the closed-loop reservoir management which proved that recovery factor would increase if there were tighter integration between short-term and long-term decisions (e.g. simulation is performed before operating settings are executed). (Jansen, J. D., Brouwer, R., & Douma, S. G., 2009).

2.2.3 Uncertainty in Instrumentation and Simulations

An accurate prediction model of reservoir condition must start with an accurate depiction of the present conditions. The equipment and instrumentation used in the industry are exposed to much wear and tear. This results in inaccuracies for data-gathering and the use of human supervision for calibration is, therefore, essential to ensure the necessary quality of input (Grimstad, Almklov, Foss, Gunnerud, 2015).

2.2.4 Disruptive Operational Events

When planning a long-term drainage and production strategies the conditions in the reservoir changes at a slow pace, and in a predictable way. However, due to disruptive operational events like equipment failure, maintenance, well-testing, and pigging, simulations portrays an inaccurate picture of the future. These events are not accounted for in simulations, if there, was a software platform where every piece of information was gathered and put into a holistic model.

Then most scheduled maintenance and other operations could be accounted for in the simulation.

2.2.5 Limitations in Software

One of the larger barriers to the deployment of IO is the lack of software standards within reservoir engineering. The absence of these standards is made evident when transitioning between software suites, by different vendors. In the current market, there are multiple types of models and simulators; these models are not compatible with each other, and proprietary files are common. One such case is that one type of software is used for data acquisition while another is used for reservoir optimization, causing a less than seamless transition. It is common for oil companies to change between the different umbrellas of reservoir management software. By doing this, there's a risk of leaving crucial information behind, such as how the data was initially processed. This meta-data is essential for a complete overview of an asset, and the information is susceptible to degradation in the translation between previous and current software platforms.

2.2.6 Trust in Models

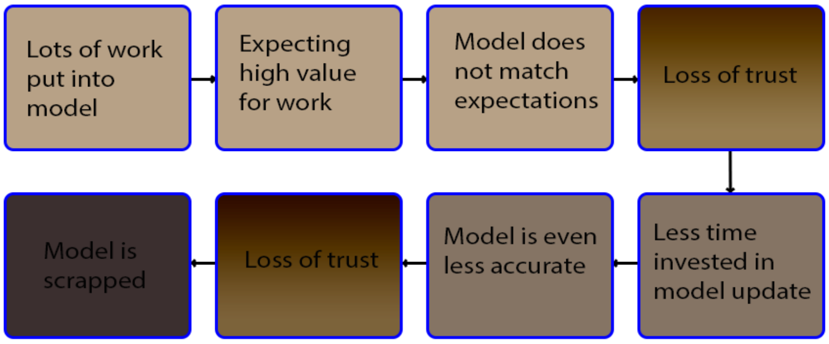

Lots of work put into model
Expecting high value for work
Model does not match expectations
Loss of trust
Less time invested in model update
Model is even less accurate
Loss of trust
Model is scrapped

Figure 2.10: The process of losing trust in a model, based on a model from On Why Model-Based Production optimization is Difficult in the Upstream Industry by (Bjarne Grimstad, Petter Almklov, Bjarne Foss, Vidar Gunnerud, 2015)

One of the difficulties revolve around trusting the output of a simulator or replica of reality. This confidence can either be increased or reduced, depending on how the output matched the expected output. It usually follows the process described in Figure 2.10 (Grimstad, et al. 2014). The key to understanding this is that every engineer has an expected outcome, not necessarily the correct numbers, but a physical understanding of what will happen during the simulation. If the simulation does not perform as expected, the user loses some of the built-up confidence to the simulator. Again, if less time is spent the outcome is as expected, even less accurate, and eventually, the model is scrapped. The simulators are usually black-boxed, the interface presented to the user only require input, and in turn produce output, not the process in-between (Figure 2.11). Lack of trust happens due to lack of insight into what makes the results the way they are. To fully trust the results there is a requirement to understand the reasoning for a simulation, understanding the output, limitations, and uncertainties within the black-box.

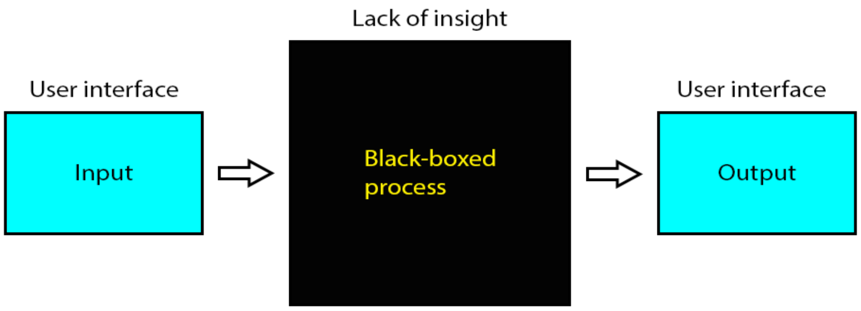


Figure 2.11: Illustration of a black-boxed simulator

There is no easy way of building trust to simulations nor models associated; it is dependent on the experience of the user. In black-boxed applications, engineers must use a method called Black-Box testing to examine the functionality of the simulator. The engineer is familiar with what the simulator should do but not how it does it, by inspecting (usually very specific) test-cases it can uncover what the simulator is able and not able to handle. This could increase the trust but never reveal exactly what happens in the simulator. By changing the fundamental principle of a simulator from a black-box approach to a transparent white-box approach (Technology Conversation, 2016), one can consider the reasoning, limitations, and uncertainties that are present in the simulator by inspecting and testing the separate parts of the source code. White-boxing a simulator itself does not solve the problem, but through more thorough white-box testing techniques, engineers can determine exact behavior. These tests can only be performed if access to the source-code is granted.

Chapter 3

Introduction to Software development

Through this chapter, there will be a review of the different two development models for software; proprietary (closed-source) and open-source. It will highlight some of the benefits and drawbacks of each type, as well as how the communities surrounding the software develop. It will also be an investigation of what lie ahead in the software industry, open-source vs. closed-source.

3.1 Framework

Reservoir engineers are dependent on complex subsurface software to have a successful development and management of an asset, from discovery to the end of production. Today, there are multiple software suites capable of performing such tasks. Saudi Aramco uses their in-house simulator GIGAPOWERS (oilandgasnewsworldwide.com, 2016), Haliburton is maintaining and further developing Nexus (Landmark, 2016), as well as introducing a new software platform called OpenEarth Community (OEC, 2016). However, there is one software suite that separates from the rest, regarding both popularity and market dominance; Schlumberger's Petrel and their reservoir simulator ECLIPSE. Petrel is a software mainly used as a platform tool to communicate complex subsurface information between disciplines and serves as a base platform for all kinds of visualization and simulation. An upcoming competitor is the Open Porous Media Initiative, which relies on an open-source development scheme; it builds on the fundamentals of open-source development and has used building blocks from other open-source initiatives to achieve success.

The principle behind software development is the process of creating, maintaining, and improving applications and frameworks to produce an application or software (Wikipedia: Software development, 2016). Software can be built for any number of purposes, ranging from computer games for entertainment, to simulators that to a large degree of accuracy depicts and predicts the real-world behavior of physical phenomenon (Wikipedia: Simulation, 2016). The development strategies are divided into two main categories; open-source and closed-source (proprietary). These two classes have different approaches as to who can maintain, solve problems, and develop the software. It is, therefore, crucial to define the two methods.

3.1.1 Proprietary (Closed-source)

Closed-source development is focused on developing, maintaining, and improving an application or a software in-house (Bestpricecomputer.co.uk, 2007). The distribution of such applications is done through compiled-executable packages; which does not allow access to the source code. The end-user does not buy the software itself; rather the end-user buys the right to use the software. Closed-source development allows for the manufacturer to protect its source code as a trade secret, by black-boxing, and enforcing proprietary file formats. A company well known for such an approach is Microsoft, the company developed both Windows and Microsoft Office in-house and had retained from distributing the source code, making the in-house developers at Microsoft the only people able to access and modify the largest office suite in the world. However, in this case, the file formats were previously proprietary but as of 2006 Microsoft Open Specification Promise (OSP) gave a none-sue agreement for the use of the file format (Mechell, 2008).

Closed-source development has a lot of benefits when it comes to the ease of creating value, through royalties and licenses. The developer can sell the right to use the software but does not need to give you access to the inner workings of the program, the source code. All the end-user experiences are the user interface (UI) created by the developer for the purpose of displaying and receiving information. This lack of insight may prove to be a challenge for the further development of the application and development of trust. Due to the lack of access, it does not enable the surrounding community to do incremental modifications to improve their workflow and in turn, contribute to an ever-evolving software. In short, closed-source is more about protecting the owner rather than enabling the end-user.

3.1.2 Open-source

When developing within an open-source environment, the end goal is to make a computer software or product where the source code is publicly available (OSI, 2016). The software is licensed out under an open-source licensing scheme such as GNU GPL (General Public License, GPLv3, 2007), where the source code is made available for public use (Table 1). The open-source approach focuses on the community and empowering everyone to develop new software, fix bugs, and collaborate to improve what already exists. Building on the shoulders of giants.

Some of the problems associated with an open-source software development are the collective goal of any project, the organizational structure, and the of realization of direct value. As you do not sell the software directly, there is no direct method to tap the created value. However, as performed by Red Hat and Canonical (creator and primary developer of Ubuntu) you can sell support services. Another way of doing this in parallel to the gaming industry, it could provide a free software platform, but offer plugin support services for the implementation to a platform. Making for direct incentives for participants in the community to further build upon what has already been created. In a complex project where there are hundreds, and even thousands of contributors it can be a hurdle to get the project moving in a common direction. There could be contributors who strive in different directions, and since there usually is a lack of organizational structure surrounding the project, there are difficulties incentivising contributors to do specific tasks. Sometimes the challenges are enough. However, there needs to be leadership in place, which task is to guide the project towards completion. As illustrated by Linuxs development, there are benefits of having a broad set of creative contributors.

Linuxs development was the inspiration of the Linux development model (Narduzzo, Alessandro, Alessandro Rossi, 2008). Linus Torvalds, the creator, performed the feat of creating an operating system built upon open-source development. He found that by admitting that the best ideas were not necessarily his own, rather, the ideas of contributors allowed him to expand and implement functionality that would never have seen the light of day (Raymond, 1999). Linus Torvalds was not necessarily a creative genius; rather, he was an excellent coordinator, he managed something nobody at the time thought was possible.

Therefore, Linuxs development was for many people a milestone, which illustrated that complex software solutions could be developed if enough contributors are allowed to participate. Or as the book *The cathedral and the bazaar* (1999) phrases it; Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone.

3.1.3 The Onion Model of Open-Source and Proprietary Communities

When describing a community within an open-source environment, there is often references to the onion model (Figure 3.1, Nakakoji et al. (2002)). It represents the community as an onion where the most influential roles are in the center and decreasing the degree of influence towards the outer rim. The passive user is the largest group, which consists of those who only use the software (Nakakoji et al. (2002)). This figure provides insight into what roles are offered in both an open-source environment and a proprietary model.

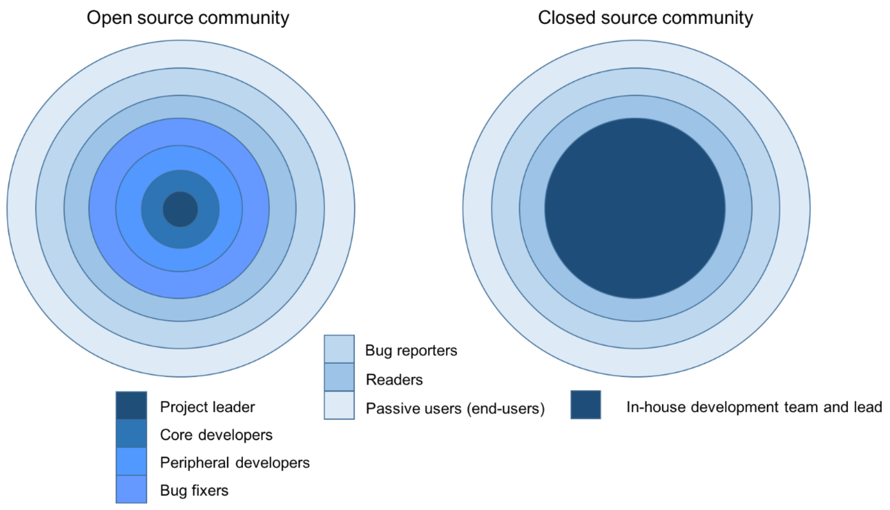


Figure 3.1: The onion model of an open-source and proprietary software inspired by Nakakoji et al. (2002)

In the proprietary model, bug-fixing must be performed in-house, and cannot be influenced by the end user. Therefore, the developer must prioritize, as opposed to the open-source model, where you are free to offer solutions yourself, and the developer needs less time for the problem to be addressed. Therefore, critical issues can be resolved without having to wait for the developer to prioritize.

By skeptics, the open-source communities could be described as anarchistic by nature, where everyone can contribute and mess up the existing framework. This is not the case for all projects; in OPM, there is a central leadership involved that has the necessary authority to set general direction and delegate tasks. The project lead also has the power to either accept or discard solutions, new features, or patches. Therefore, it is not straightforward to intentionally sabotage a project. The idea behind an open-source community is the thought that everyone is equal, but this trust can be broken and the user or developer banned from participating if content is in direct violation of terms of use.

3.1.4 A Move Towards Open-Source Development

Historically most software released was in the proprietary and closed-source format. This distribution type was done to protect the developer and distributor from competition and exposure of trade secrets. As the computer grew from its infancy and the applications have grown in complex, the need to keep trade secrets have decreased, in favor of delivering the best and most agile software solutions. Open-source software offers a competitive edge in flexibility, and the end-user is utilized completely different. Per BlackDuck (2016), an open-source security company, there is a clear sign that in many cases the development has shifted from closed-source to open-source. BlackDuck conducts surveys among software developers and users. The study tries to unravel the trends going on in the software industry, and their last survey conducted in 2016 identified some of the features that enable open-source software (OSS) to increase market share. The three biggest features are 1. The quality of Solutions, 2. Competitive features & technical capabilities, 3. Ability to customize and fix (BlackDuck Software, 2016). The organization has also identified that 65% of all companies that partook in the survey uses open-source software, up from 60% the year before.

3.2 Licensing and Associated Business Models

All software that is commercially available is protected by license schemes. Several schemes protect the software, some favoring the creativeness of the user; others favoring the interest of developers. All license types allow for certain business models. There will first be an introduction to some of the major licensing types with coherent examples, followed by analysis of some of the business models associated with the licensing schemes.

3.2.1 The Spectrum

There is a broad range of license types (Figure 3.2). They all follow an axis from almost all rights in the software being retained, as a trade secret, on one end and the other end with public domain, there is right to use, distribute, and even sub-license the software.

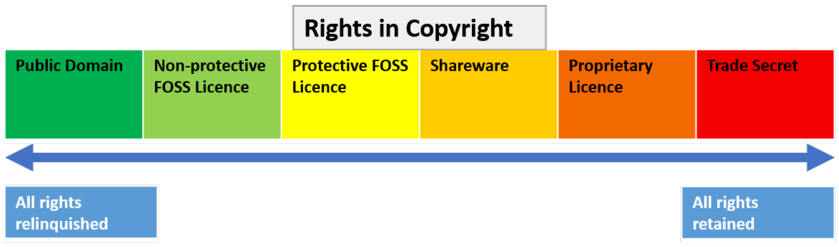
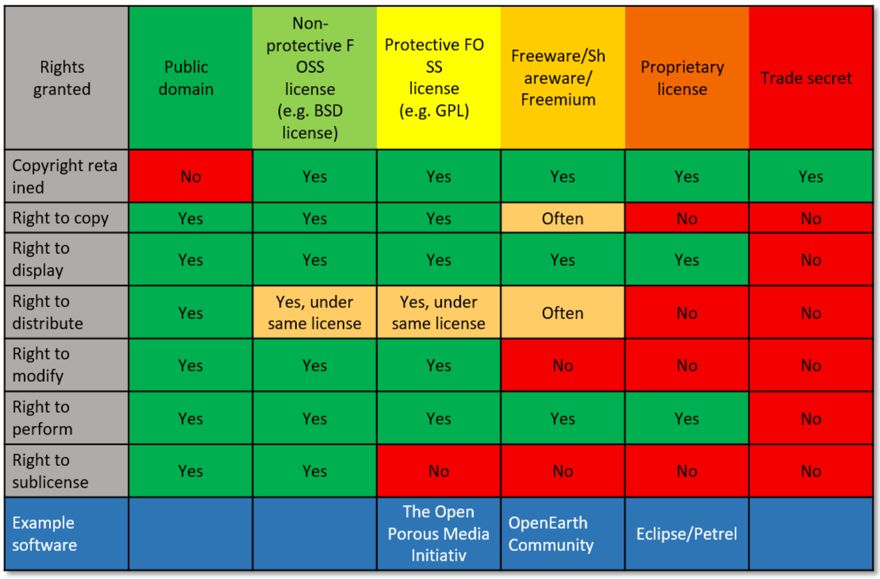


Figure 3.2: Rights in Copyright, based on Mark Webbinks definition of licenses in use. Left-side of the axis favors the rights of the end-user while the right-side favors the rights of the developer. (Larry Troan, 2005)

The most commonly used licenses are the ones stated in Figure 3.3. Eclipse is licensed with the under a proprietary license, while the Open Porous Media Initiative is licensed under GPLv3. The OpenEarth Community would be somewhere in the middle, due to the lack of information about the actual licenses it is impossible to make a definite evaluation. While the license-holders might retain the source code on the right side, they may provide development tools for development through their Application Programming Interface (API). API is essentially a high-level programming language or interface that allows for the use of built-in functionality within the software environment. By distributing this interface, a developer can enable the development of additional functionality by out-house developers, without exposing the source code.

3.2.2 Proprietary Licenses

Proprietary licenses give the distributor the intellectual property. It is usually associated with closed-source development, although exceptions do exist. By buying a license, you have the right to display and perform the software (Table



Rights granted	Public domain	Non-protective FOSS license (e.g. BSD license)	Protective FOSS license (e.g. GPL)	Freeware/Shareware/ Freemium	Proprietary license	Trade secret
Copyright retained	No	Yes	Yes	Yes	Yes	Yes
Right to copy	Yes	Yes	Yes	Often	No	No
Right to display	Yes	Yes	Yes	Yes	Yes	No
Right to distribute	Yes	Yes, under same license	Yes, under same license	Often	No	No
Right to modify	Yes	Yes	Yes	No	No	No
Right to perform	Yes	Yes	Yes	Yes	Yes	No
Right to sublicense	Yes	Yes	No	No	No	No
Example software			The Open Porous Media Initiativ	OpenEarth Community	Eclipse/Petrel	

Figure 3.3: Software licenses and rights granted in context of the copyright according to Mark Webbink (Larry Troan, 2005)

1). License, serial keys, or other types of verification limit the use of the software; restricting the use of additional computers, reducing the scalability of the software.

3.2.3 Proprietary Business Model

The most commonly used software in reservoir engineering and other geosciences is Schlumbergers software platform Petrel. Petrel is a multidisciplinary software platform that allows for subsurface disciplines to share valuable information about a reservoir. It prepares files and displays the results from the reservoir simulator Eclipse.

The software suite is distributed through a licensing scheme, which allows for one computer per license. These licenses are expensive, although some price discrimination is present, they are subscription based estimating one license costing one million NOK per year. By purchasing one of these licenses, you get access to all of Schlumbergers features and software. This is a somewhat controversial scheme, as most of the E&P companies subscribing to Petrel only use a fraction of the software available. In other words, you pay for a higher amount of applications than you will ever use.

Schlumberger does not provide access to the source code of any of its PETREL based applications, rather, it distributes an API that allows for the independent development of petrel-dependent plugins. These plugins can be distributed through a marketplace called Ocean.

To get the plugin to the market, the Ocean store, Schlumberger needs to review and make sure that the plugin fulfills the terms of use. This process of evaluating, allows Schlumberger to get exclusive first-hand knowledge about it, and if it is in any way revolutionizing they could potentially buy the plugin. By doing this, Schlumberger maintains a dominant position in the reservoir software market. If the plugin is purchased, Schlumberger may include it as a base feature in the next release iteration of Petrel, making the software even more complex. This business model saves costs and allows for Schlumberger to have third-party developers, making the public and other major oil and gas companies act as developers for Schlumberger. Even if the application is not bought, Schlumberger always gets an unconfirmed share of the Ocean marketplace price.

An example of this practice is the story about Shell and their tailor-made applications. As a part of Shells practice, there is significant investment tied to making specific applications for very niche markets within the company itself. They make these applications tailored to the Petrel platform, however, to utilize and maintain the application it requires significant operational costs Shell made a deal with Schlumberger allowing for them to have the applications and supporting it within the Petrel ecosystem, free of charge for Shell. Schlumberger, in return, got to sell this plugin to all oil and gas companies, giving Schlumberger significant gains without having to invest in the development of the software itself, allowing Petrel to expand in functionality.

3.2.4 Open-source Licenses

The Open Porous Media Initiative is licensed under the GPLv3 scheme. It is the same license that is used for the Linux kernel; the user has the right to copy, display, distribute, modify and perform the code. The GPLv3 ensures that the fundamental rights of freedom in open-source development is maintained in each build. GPLv3 also ensures that modifications and other builds that will be distributed are kept under the same license. Thus, no software can be based on the open-sourced software and then suddenly decide to retain the fundamental freedoms that are described in Table 3.1. However, something that should be highlighted is the fact that you can base your in-house modification on the GPLv3 licensed software and not distribute it, if it is kept in-house it does not have to be published.

3.2.5 Open-Source Business Model

One of the difficulties in open-source initiatives is the way to make a sustainable business model that can turn a profit. As Open Porous Media has not yet been released as a commercial product, rather a product in development; this section is not confirmed and presents general business models in open-source development, drawing inspiration from other similar value chains in the software industry.

The fundamental idea of OPM is to make a simulator that stimulates open innovation and reproducible research for modeling and simulation of porous media processes (OPM, 2016). It is a capability platform that allows for any further development of porous media processes. The focus is mainly on reservoir simulations. However, there are exceptions of applications outside this very specific area such as CO2 storage. The idea of the initiative is to allow for every willing individual to contribute, as OPM itself is based on other open-source initiatives such as DUNE (OPM, 2016).

The OPM platform contains several applications. Flow, the fully implicit three-phase black-oil simulator; Upscaling, the tool used for upscaling flow-based permeability, relative permeability and capillary curves; ResInsight, a powerful tool for visualizing reservoir simulations (OPM, 2016). All mentioned applications and their related source code are released under GPLv3, ensuring free license in a long-term perspective.

Regarding business model, there are multiple options. Any of the following business models could be combined to tailor-made solutions for customers. The common way of distributing a software developed with the intention of being open-sourced is the Red Hat model. Red Hats software is available on their website;

Table 3.1: The four fundamental freedoms of GPLv3, http://www.gnu.org/licenses/quick-guide-gplv3.html

1.	The freedom to use the software for any purpose,
2.	The freedom to change the software to suit your needs,
3.	The freedom to share the software with your friends and neighbors, and
4.	The freedom to share the changes you make.

you are free to compile and install it at your discretion. Also, Red Hat sells support services. By subscribing to their support subscriptions, they make it easier to install, and if you are having trouble they give you the help you require.

An alternative could be to use a model called Software-as-a-Service (SaaS), this is a general approach where the service is centrally hosted, in e.g. a cloud service, and people use the services on-demand. This means that OPM would rent server capacity in a cloud service and OPM receives payment through an additional profit margin (Figure 3.4). By hosting the service in a cloud computing environment, one can access the service from anywhere. The price discrimination could be based on several factors; the number of users; the number of processors required; the amount of computing time; or a subscription based support model. This means that the initial setup price for the software is lower than for a proprietary software where you must pay up-front for the license (in addition to supporting services). There are multiple benefits of using the SaaS model. As the software is centrally hosted, there is no need to support older versions of the software, as there is only one version. Patches are applied at the developers discretion, and could, therefore, be updated quicker than a similar software hosted on the computers of the users, which require manual updates.

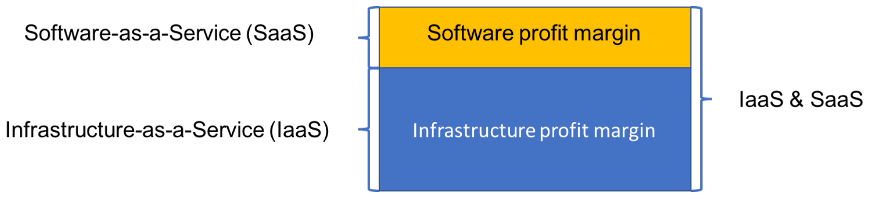

Software-as-a-Service (SaaS)
Software profit margin
Infrastructure-as-a-Service (IaaS)
Infrastructure profit margin
IaaS & SaaS

Figure 3.4: Potential profit margins in a cloud-based environment, IaaS & SaaS, potential profit in the y-direction

A third alternative could be the use of a model called Infrastructure-as-a-Service (IaaS), in this model one sells the computing capacity for the application. One such example is the Microsoft cloud, Azure. It is a general approach where companies can install their software in the cloud and subsequently use the computing power to generate the results. As this model revolves around the computer capacity, not the software itself, this business model is to be seen in context with SaaS.

OPM could host virtual machines on any laptop or desktop, connected through e.g. SSH to their servers for processing. This means that E&P companies, especially smaller ones, do not have to purchase high-end hardware to run the simulations. OPM could design server capacity for peak usage, and quickly add additional racks if necessary and as the customer base increases.

OPM would mainly focus on the smaller upstream companies who do not have the sufficient capital to invest in hundreds of ECLIPSE licenses, as the costs associated are high. However, by providing support services to smaller companies, the companies do not have to have to invest in dedicated IT department for maintaining and developing their own software solutions. As opposed to PETREL, which contains hundreds of programs, OPM would offer the software on a need-to-use basis; you pay for what you need; not for all the unnecessary features.

Both Eclipse and OPM uses the same file formats. These formats are the proprietary rights of Schlumberger, however, due to interoperability, it can be utilized by OPM. By doing so, OPM concedes that there is one dominant file standard, and therefore embraces an industry-wide standard rather than inventing their own. This would aid in a transition phase, as there would not be much alterations in the workflow that are already in use. The main difference would be the simulator itself and the access granted.

3.2.6 Vendor Lock-In

If only proprietary software is used and the interface of the applications themselves are only compatible with other software offered by the same developer, there is a significant danger of becoming too dependent on one software provider. This is called vendor lock-in. If you find yourself in this situation, it is common for vendors to increase prices for the same services. This can be avoided by using several software suites, the drawback, however, is that the software suites may not be inter-operable with each other.

An example of this practice is the partnership between Schlumberger and several technological universities. They offer free licenses for educational purpose. However, the co-effect is; when the student has graduated, Schlumbergers software is the only software that the engineer has utilized, and naturally want to continue using in the industry. Therefore, it should be an obligation of the University to enlighten students about alternatives, opening their eyes for different software.

3.2.7 Reservoir Simulation and Standardization

In reservoir engineering, there are large amounts of data gathered. Optimal decision-making is dependent on the optimal usage of this data. Illustrated in Figure 3.5 is a description of the dependencies within the domain of reservoir engineering. The level of autonomy decreases as you move upwards through the pyramid. This pyramid is related to the process of optimizing production and calibrating already existing fields.

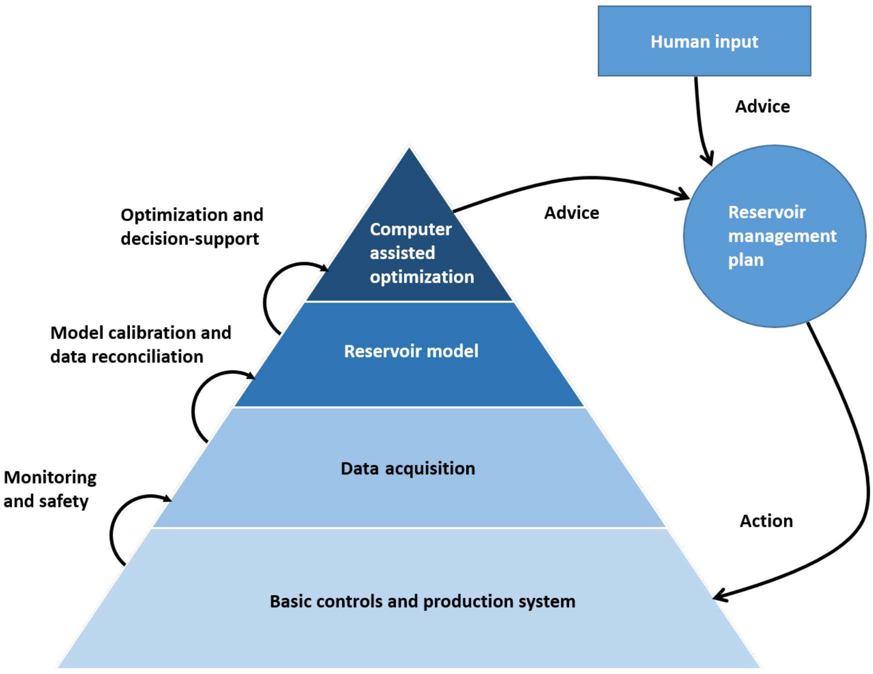

Human input
Advice
Optimization and decision-support
Computer assisted optimization
Advice
Reservoir management plan
Model calibration and data reconciliation
Reservoir model
Data acquisition
Monitoring and safety
Action
Basic controls and production system

Figure 3.5: Top-to-bottom dependencies in reservoir engineering, inspired by Grimstad, et al. (2014)

Basic controls and production system is the most basic level. The wells, valve controls, and all the infrastructure used for transportation of reservoir fluids. It monitors the output of the wells. This could be automated in such a way, that it could be reliant on computer input and output, and the actions predefined in the reservoir management plan.

Data acquisition is the instrumentation level of the pyramid. It is where all temperature, pressure, and multi-phase flow meters gather information and stored. The dependency on the production system is large, it monitors whether expect output is the actual production, it allows for trend-based tracking of parameters of the reservoir through the production system. Also, it oversees the field to ensure safe operations. The sensors and their instrumentation value might not always be correct; production engineers, therefore, must label the data according to a quality scale. This reduces the degree of autonomy applied to this level, as it requires human supervision.

The reservoir model level is then fed with the information from the instrumentation level. This level is used for calibration of current condition in the reservoir. In a process called history-matching. The reservoir is updated and matched with all available information until the reservoir model closely matches past behavior (SLB, 2016). The reservoir model is then compared to the predictive model that the previous updated reservoir management plan is based upon, to check whether the instrumented output is the same as the predicted one. If this is not the case, the reservoir model must be re-calibrated, and in turn, account for the changes that have happened since last time. Several things could happen with the reservoir that does not get accounted for in the initial predictive plan especially things like unforeseen maintenance, equipment failure, undetected faults, and sealing barriers inside the reservoir.

The uppermost level of the pyramid is the computer-assisted optimization as changes have been calibrated in the lower level of the pyramid, and a history-matching has been performed. The problem remains to find an optimal path forward and an optimized reservoir management plan, with regards to placement of infill wells, drainage strategy, and the volumes that are produced, etc. This level also must account for unexpected events and evaluate what action should be taken to optimize production.

All these dependencies can be found in different fields of engineering, however, for reservoir engineers, it is especially cumbersome due to the significant amount of uncertainty of a subsurface environment. The end-goal of IO is to automate much of these processes to move reservoir engineers from iterative tasks to value adding activities. If you can change the abilities of reservoir engineers to utilize the software as a tool for decision-making, the upstream industry can better cope with fewer engineers.

In order for this development to take place, there is a need for an increase in both trust in the output from simulators and the ability to modify the simulator itself. An open-source approach might lead to increased trust and allow for modifications, but the output also have to be accurate - on par with the current standard for reservoir simulators. In the next chapter, a benchmark test has been performed, Flow is compared with the reference simulator, Eclipse. It will illustrate the current reliability of the open-source reservoir simulator, and test the competitiveness of the solution, Flow version 2017.04.

Chapter 4

Norne Benchmark Study

The domain of reservoir simulation is complex, both the framework used to calculate behavior and the data used for making the model contains significant amounts of uncertainty. The data used to calibrate the model come from secondary (production data, seismic, and geostatistical analysis) and direct (core data and petrophysics) sources, thus, many areas of expertise are involved in the making of a sophisticated and comprehensive reservoir model. Throughout the lifetime of a reservoir there is continuous process of improving simulations to fit the real life conditions, to better predict the future. In today's market there are several capable reservoir simulators, the most dominant in terms of market share is a simulator called Eclipse 100. Eclipse 100 is a black-oil simulator that the community of reservoir engineers have endorsed as the reference simulator. In this chapter, Eclipse will represent the proprietary development model and the reference, while Flow, delivered by Open Porous Media Initiative's reservoir simulator, is the open-source alternative. It will test whether altering the fundamental principle development model; it will increase flexibility without sacrificing reliability. The hope is to offer operators and engineers the possibility to adjust and modify software to fit their needs.

Benchmark tests are necessary; it gives an indication of the validity of the output and the performance of the computational framework. For Flow, the results have to be trustworthy, and indicate a good match to the reference in order to competitive in current market. The benchmark test uses data from the Norne full-field reservoir model. The reservoir model builds on the knowledge acquired from a producing reservoir found at the Norwegian Continental Shelf (NCS). It is one of very few full-scale reservoir models available for students and researchers.

The exercise is designed to test and compare the accuracy and the computational performance of Flow. The simulations were run using the same hardware and operating system, with as few programs as possible interfering the simulation procedure. This identical setup ensured an objective basis for comparison.

The chapter is structured like a report. It starts with an introduction and a description of the reservoir. There is a section on the methodology utilized for the benchmark study. The results and interpretation section contains processed data and the initial analysis. And the chapter rounds off with a brief discussion of the results and a conclusion.

4.1 Introduction

Norne is a complex reservoir, some of the features that the simulators have to handle are: Dissolved gas, vaporized oil, transmissibility multipliers, pressure-dependent porosity and transmissibility, end-point scaling for relative permeability and capillary pressure, history-matching production well controls that change throughout the simulation schedule (OPM, 2017). The combination of all these features makes the test tough, even for commercial simulators.

The simulation and post-processing were performed using a majority of open-source software. Except for Eclipse, Schlumberger Simulation Launcher, and a VPN client. The licenses associated with the open-source software utilizes the fundamental freedoms of GPL (Table 3.1). The intent was to test the availability and the option of using open-source software in the domain of reservoir engineering.

4.1.1 Description

In 1991, Well 6608/10-2 discovered a hydrocarbon reservoir located inside License Block 6608/10 and 6508/1, 200km off the coast of Norway. The wildcat found an oil column of 110m with an overlying gas cap of 25m. At the time of discovery, it represented the largest oil discovery of the decade on the NCS. (Steffensen and Karstad, 1996). This reservoir was named Norne.

Norne reservoir model and production data were handed to the IO-centre through an agreement between the license holders of Norne (Statoil, Petoro, and Eni), as a part of a benchmark study conducted in 2006. The case has since proven to be instrumental in research and education. Giving students and researchers access to data sets based on actual production data.

Reservoir Geology

Norne consists of five different formations. From top to base these are Garn, Not, Ile, Tofte, and Tilje. (Tables B.1 and B.2) (Dalland et. al. (Eds.), 1988). The producible parts of the reservoir consist of sandstone with a large variety in grain-size. The impermeable *Not* formation separates the Norne field into two isolated parts, the above and under *Not* sections. The above *Not* structure includes the Garn formation, a sandstone of coarse grain size, initially saturated with gas. The under *Not* structure includes Ile, Tofte, and Tilje formations. Ile and Tilje contain reservoir wide interbedded shale and silt layers which drastically alter vertical flow properties.

Initial Drainage Strategy

Vertical Flow Barriers

The initial drainage strategy was to increase the height of the water-oil contact (WOC). The WOC would be raised using water and water-alternating-gas (WAG) injectors. These would distribute water underneath the oil column throughout the reservoir, in an attempt to boost the level of the WOC evenly. A combination of sea and production water would be pumped into the reservoir to achieve this effect (Steffensen and Karstad, 1996). The purpose was to ensure an efficient volumetric sweep. The reservoir geology, (Table B.1, B.2 and Figure B.1) was not particularly suited for this strategy, as they contain interbedded shale and silt layers. Through the early stages of production, and as the development team learned more about the reservoir and the vertical properties of the formations it affected the initial strategy. This knowledge encouraged the production team to initiate a flanking-injection maneuver to increase the volumetric sweep. The idea of a flanking maneuver is to use water injectors located in the outer parts of the reservoir, to displace hydrocarbons towards the production wells located in the center.

Horizontal Flow Barriers

Figure 4.1 illustrates the faults inside the reservoir. It shows a significant degree of compartmentalization. These faults have a direct impact on the drainage

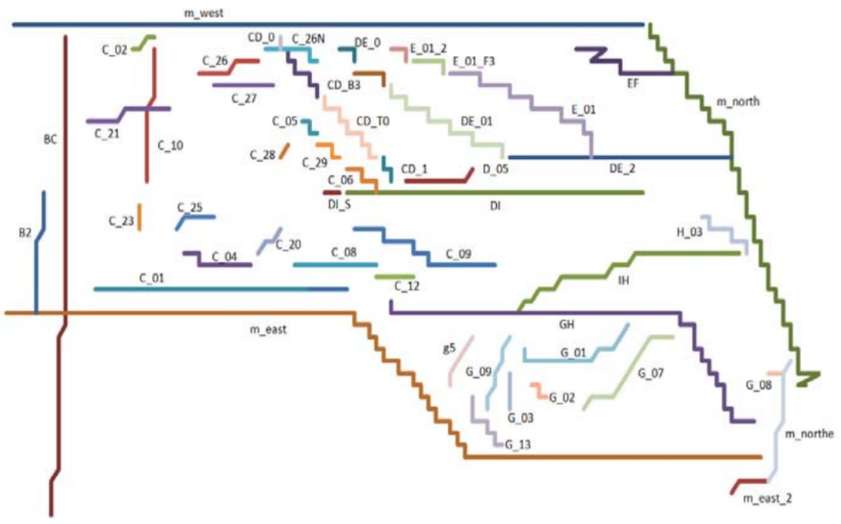


Figure 4.1: Schematic illustrating the faults applied in the Norne reservoir model (Morell, 2010)

strategy and the placement of the wells. It means that water injection-flanking maneuvers are difficult to implement with full effect, as the horizontal barriers inhibit the displacement of hydrocarbons. The drainage, therefore, has to adhere to the compartmentalization of the reservoir. It is a sub-optimal situation, where the reservoir will need multiple wells per compartment to maintain pressure and produce with best possible efficiency.

Several methods allow for the detection and mapping of these faults. These include seismic measurement, petrophysics, well tests or production data, which can be used both independently or in a combination. The identification of faults is a time-consuming task with lots of uncertainty; however, the real challenge is to estimate transmissibility and diffusivity of the semi-permeable faults, as they have a significant effect on the total production.

4.1.2 Sensitivity Study on Norne Reservoir Model

Sensitivity analysis is a study of the variables that both, directly and indirectly, relates to the results of the simulation. Saltelli et al,. (2004) proposed a definition: "The study of how uncertainty in the output of a model (numerical or otherwise) can be apportioned to different sources of uncertainty in the model input.” In short, it tries to establish a relationship of the input to the output of the simulation. A sensitivity analysis investigates whether a change, of a predetermined magnitude in one parameter, correlates to a change in the output from the simulation. It is an attempt at uncovering whether several inputs may lead to the same output or if the simulation is unique. Sensitivity analysis is a tool that aids engineers to increase their understanding and intuition of the data associated with the simulation. It also establishes what parameters framework of what variables are sensitive and insensitive to change.

Sensitivity analysis uses ensemble where every realization contains a small alteration in one input parameter, to investigate whether there is a change in the corresponding output. Flow was designed specifically to do realizations of large ensembles. Therefore, performing a sensitivity study would be an adequate way of evaluating both the single core and dual core performance.

The benchmark test is, therefore, an exercise, designed to investigate whether Flow can handle some of the typical workflows of reservoir engineers in regards to performance and accuracy. The exercise is not intended to accurately history-match the Norne reservoir, rather, the end-goal is to check the run-time and validity of the output. These results will then compare with the results from Eclipse, as the reference simulator. The benchmark study will investigate the validity of the results concerning field production data; well production data; well pressure data; saturation of oil, gas, and water; and static properties.

4.2 Method

As introduced earlier in the chapter, the Norne reservoir model is a highly-compartmentalized reservoir with a significant number of horizontal (faults) and vertical (impermeable layers) flow barriers. Therefore, a typical workflow would involve a sensitivity study of transmissibility and permeability through these flow barriers. This would be done by changing the properties associated with the layers and faults, and checking the results towards the production data. In the Eclipse input deck flow barriers are inserted through keywords. These keywords define the properties, geometry, and direction related to both faults and layers.

Flow and Eclipse use the same input file format, namely the Eclipse input deck. The input deck use keywords to distinguish between parameters, properties, and geometry of the reservoir. These keywords allow for detailed description of individual layers and the properties therein. The input deck contains keywords which give the option of testing parameters before implementing them to the final model. Through keywords like ”MULTFLT” and ”MULTPLY” it is possible to modify the initial values and create ensembles.

Keyword MULTFLT

MULTFLT modifies the transmissibility through faults. It utilizes the previously defined faults in the keyword FAULTS. FAULTS contains the geometry and direction of the fault (Schlumberger, 2016). MULTFLT is a factor between [0, 1] multiplied with the transmissibility in the direction of the fault. Table C.1 contains the initial values (IV) given with the Norne reservoir model.

Keyword MULTIPLY

MULTIPLY is a keyword that allows for specific modification of properties in three-dimensional regions set by the user. In this case, the vertical permeability simulates the low permeable areas inside and between the formations.

PERMZ (Table C.2) is the parameter to adjust the value of the permeability in the Z direction; and is multiplied with the original permeability. This multiplication causes the effective permeability to either increase or decrease. For the Norne reservoir model, these layers run through the entirety of the reservoir and are therefore highly sensitive to alteration. Table C.2 contains the default PERMZ initial values. The table shows, as already mentioned, significant differences in the vertical permeability through the entire reservoir.

4.2.1 The Ensemble

For this exercise to take place, it was necessary to create an ensemble consisting of multiple realizations with varied parameters. As the goal of the exercise is to benchmark the performance and validity of the simulators, rather than the accurately history-matching the reservoir, the values were picked by multiplying the initial values with two realistic factors. Table C.3 and C.4 contain the new values. A factor of 0.5 was used to obtain a Lower Value (LV) and a factor of 2 to obtain an Upper Value (UV).

The ensemble was created using a script written for Octave, an open-source alternative to MATLAB. Scripting this task allowed for greater flexibility in terms of editing and implementing the values in the ensemble.

TUNING

Tuning is the keyword used to set the simulator control parameters. It controls all time-step variables which include maximum time-step, minimum time-step, etc. Tuning also controls the convergence criteria, truncation error, and the maximum number mathematical (both linear and Newton) iterations, in any single time-step. (Schlumberger, 2014.1)

Tuning settings allow optimization of the simulation, in regards to run-time. This keyword is not yet implemented in Flow and uses another interface for input. Eclipse could, by optimizing the tuning settings, perform the calculations more efficient than the following results show. Optimizing a simulation in terms of tuning settings requires intimate knowledge of the behavior of both the reservoir model and simulation. Thus, for this benchmark test, the user is assumed to be inexperienced and unfamiliar with the reservoir.

Through this benchmark test, both simulators will run without custom tuning settings. This decision allows comparison between Flow and Eclipse simulators on an even playing field. Any TUNING keyword would, therefore, be excluded from the simulation and automatically replaced by the simulators default values.

4.2.2 Running the Simulation

The simulations were performed on a desktop running a Linux system, with the specifications summarized in Table 4.1. The simulations were launched with scripts in sequential order.

Table 4.1: Computer specification

Component	Specification
Operating system	Ubuntu 16.04_x 86_64
Central processing unit	Intel(R) Core(TM) i5-4670K @ 3.40GHz (no hyper-threading)
Storage	Samsung SSD 850 EVO 500GB
Memory	16 GB RAM @ 2333MHz
Network adapter	Killer e2200 Gigabit Ethernet Controller

The simulations were all ran without interference from other programs, except the VPN-client Cisco Anyconnect. The VPN-clients purpose was to obtain the license information from the Schlumberger license server located at the NTNU, Faculty of Engineering. The software that was used for post-processing the results are listed in Table 4.2.

Table 4.2: Software utilization

Task	License type	Software
Scripts	Open-source	Octave and Python
Result processing, visualisation	Open-source	ResInsight with Octave plugin Python module MatLibPlot
Result processing, data handling	Open-source	Open Office

4.3 Results and Interpretation

The result and interpretation section presents the results according to priority. First, the validity of the results from the simulation will be investigated. The validity is paramount; it is the pillar of every development decision. Comparing the output from Flow with the output from the reference simulator, Eclipse enables an investigation of the validity of the simulators. The ensemble has also been processed and will give support to the decisions that are made. The second part will be a comparison of the simulators run-times for different computational setups of Flow. This part allows for a thorough insight into the single and dual-core performance of Flow. Also, this would give an indication of the scalability of the reservoir simulator, and Flows ability to perform parallel processing.

Due to the sheer amount of available data, and to grant the benchmark test the thoroughness it requires, all raw production graphs, pressure profiles, and saturation maps, are attached in Appendix B to E. Included in this section, however, are treated data that enables easy visualization.

4.3.1 Validity Framework

When working towards verifying results, it is important to have an unbiased framework. This framework is based on the definition of relative change (Equation 4.1).

$$\text{Relative change}(x, x_{\text{ref}}) = \frac{\text{Actual change}}{x_{\text{ref}}} = \frac{\Delta_r el}{x_{\text{ref}}} = \frac{x - x_{\text{ref}}}{x_{\text{ref}}} \tag{4.1}$$

where x is the value of variable produced by Flow and x_{ref} is the value of variable produced by the reference, Eclipse. Throughout the following section, when talking about deviation Equation 4.2 defines the term.

$$\text{Deviation [\%]} = \text{Relative change}(x, x_{\text{ref}}) \times 100 = \frac{x - x_{\text{ref}}}{x_{\text{ref}}} \times 100 \tag{4.2}$$

Analyzing the ensemble involves using the standard deviation and means of the difference between Flow and Eclipse, defined in Equation 4.3, 4.4, and 4.5.

$$x_{\text{diff},n} = x_n - x_{\text{ref},n} \tag{4.3}$$

where x_{diff} is the difference in value of variable produced by Flow and Eclipse and n is the Case number.

$$\bar{x}_{\text{diff}} = \frac{\sum_{n=1}^{N}(x_n - x_{\text{ref},n})}{N} \tag{4.4}$$

where $\bar{x}_{\text{diff}}$ is the mean difference of the cases in the ensemble and N is the total number of cases in the ensemble.

$$x_{\text{stdev}} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_{\text{diff,n}} - \bar{x}_{\text{diff}})^2} \tag{4.5}$$

where x_{stdev} is the standard deviation of the differences between the cases in the ensemble.

It is always a question whether or not to use dimensionless numbers, like deviation, for comparison. The alternative would be to use the absolute values of difference. There are drawbacks of both methods, relative change or deviation are extremely sensitive when the production volumes are small, but give good indications for numbers that otherwise would be hard to interpret. The absolute difference would present the reader with the value of the differences but is hard to understand when given as a stand-alone figure. A choice was therefore made to go with relative change, because of the amount of data that would have to be presented to make sense of the absolute difference.

This framework gives a set of tools that will aid in the final analysis (Table 4.3). One assumption is that results are inherently uncertain and difficult to measure against their true value. This section, therefore, is based on comparing the results with Eclipse as a reference, using the same input and expecting the same output. It is impossible to say whether Eclipse or Flow produce the best representation of the true value. However, Eclipse is by the majority of the reservoir engineering community accepted as the reference simulator. Thus, any other simulator has to use it as a basis for comparison.

Table 4.3: Framework for verification

Deviation	Definition	Description
<5%	Insignificant	Good match
5-10%	Minor significance	O.K. match
>10%	Major significance	Poor match: Attempt at finding an explanation

For a verification of results, the simulators will simulate the Norne reservoir model without alterations. This unmodified state means that the reservoir will simulate behavior in history controlled mode from start-date (06-11-1997) to end-date (01-12-2006). This test is not predictive, it looks at the historically produced rates from the Norne field and simulates the reservoir behavior accordingly. These rates are located in the schedule file and are based on the actual production values from the production facility at Norne. The unaltered state is the case that will be analyzed in detail, the ensemble realizations will be used as support. In addition, the ensemble with altered parameters will check whether deviations are systematic or caused by chance, through the standard deviation. The cases in the ensemble are analyzed and compared to one-another in a case-by-case fashion. This will simulate the workflow that would be analysed through a sensitivity study.

The results are presented in the following order: Starting with the field-level variables, these variables are the least sensitive to change. Field-level data consists of the sum of all wells and change, although, even if everything looks equal between Flow and Eclipse at this level, differences may hide in the data for individual wells. That is why the next step is to examine the well data; these parameters are highly sensitive to changes in the reservoir and uncertainty in calculations. The Bottom-hole Pressure (BHP) is the most sensitive variable, due to the direct coupling with geometry, statics properties, and rates. Lastly, the saturations profiles and static properties are presented.

4.3.2 Validity

Field Data

The keyword RESV controls the production rates of oil, gas, and water. RESV is interpreted as the target rate of which the well will attempt to produce at.

All wells regulate the bottom-hole pressure to match these rates. The keyword only takes into account the volumetric flow of all fluids (collectively) at reservoir condition, as such, it does not differentiate between the production fluids. (Schlumberger, 2014.1).

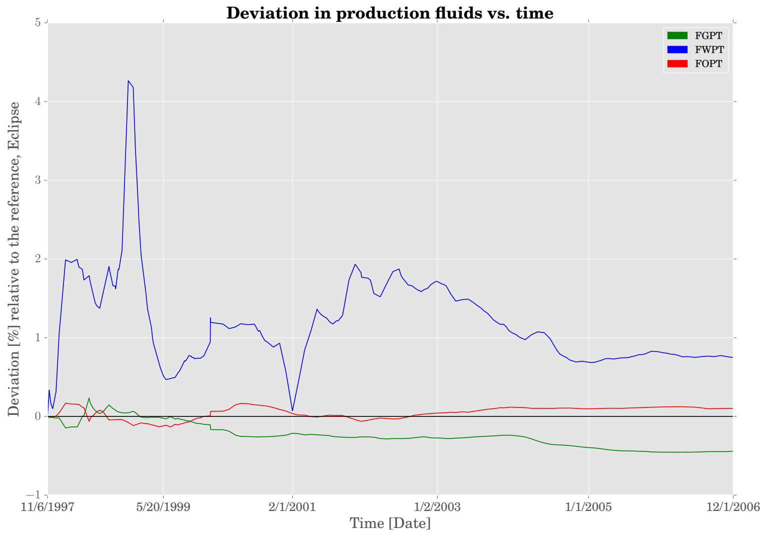


Figure 4.2: Difference from Eclipse in terms of oil [red], water [blue] and gas [green] through every report step. Eclipse as reference

Table 4.4: End difference as a percentage of the total cumulative production. Eclipse as reference

Fluid type	Cumulative production: Eclipse	Difference in production values	Relative difference
	[SM3]	[SM3]	[%]
Oil	$6.73 \cdot 10^7$	79000	0.12
Water	$2.32 \cdot 10^7$	174 100	0.75
Gas	$14.64 \cdot 10^9$	$-6.5 \cdot 10^7$	-0.44

Looking at the total production of oil, at surface conditions (Figure D.2) the curves for the total production of all reservoir fluids look like they overlap throughout the production history. Figure 4.2 illustrates the deviation in cumulative production for all production fluids throughout the simulated history. The red curve represents the deviation in oil from the reference simulation; it reaches a peak near the end of the simulation. This peak has a value of less than one percent, as illustrated in Table 4.4 and is an insignificant deviation. The blue curve represents water; it reaches a peak early in the production history. This peak happens at a time where the cumulative water production is very low (Figure D.2). Combined with a deviation of less than 5% makes this an insignificant error. The green curve indicates the deviation regarding gas; it shows that Flow produces less gas than the reference, but with an insignificant deviation. These differences (Table 4.4) are all less than 5% and is therefore deemed insignificant regarding decisions made at a field level, and therefore a good match. The same results appear in realization of the ensemble (Figure 4.3.2, even if the parameters of individual cases are altered. The standard deviation show that the deviation in production is similar in all realizations of the model, meaning that it is a systematic error and not just one case that behaves this way. These differences do, however, highlight the fact that there is some deviation in the results which may be masked in the individual wells.

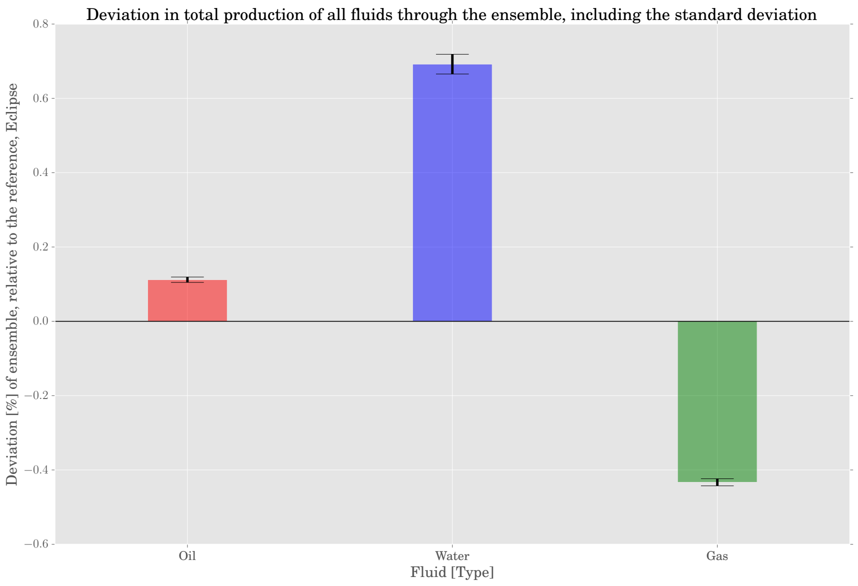


Figure 4.3: The figure illustrates the deviation in production of oil [red], water [blue], and gas [green] from the ensemble realization. The standard deviation is colored in black

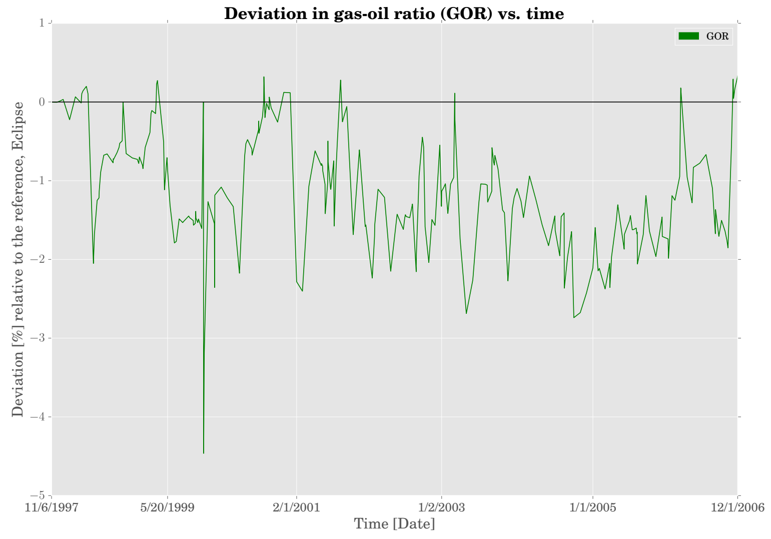


Figure 4.4: Deviation in the gas-oil ratio (GOR) [Green]. Eclipse as reference

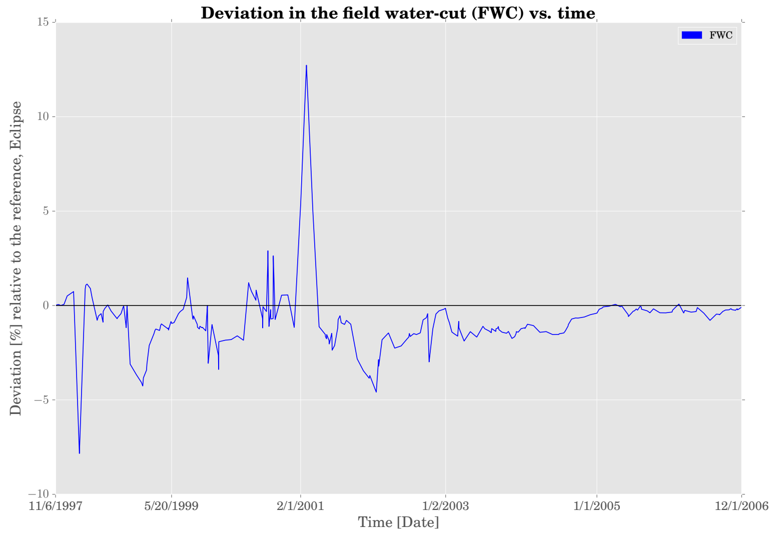


Figure 4.5: Deviation in the water-cut (WC) [Blue]. Eclipse as reference

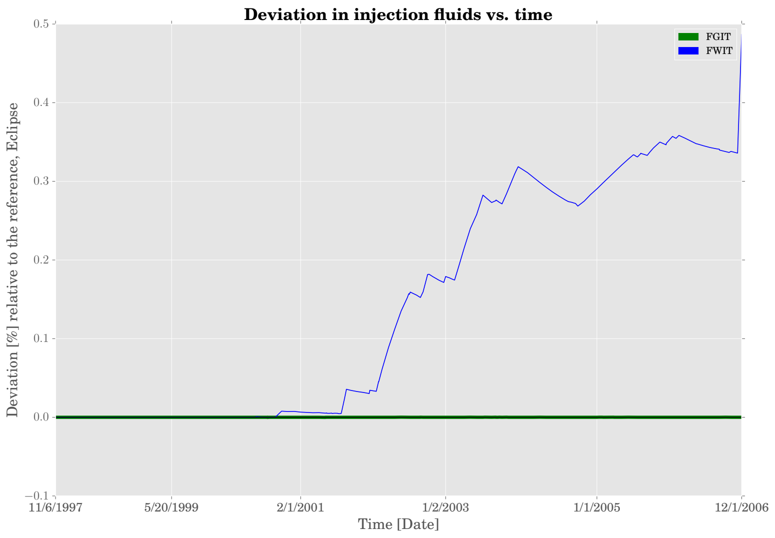


Figure 4.6: Deviation in field injection of gas [Green] and water [Blue]. Eclipse as reference

Considering the ratios, gas-oil ratio (GOR) and water-cut (WC), (Figure D.3, 4.4, and 4.5) reveals that Flow computes slightly lower average GOR and WC compared with Eclipse. The gas-oil ratio (GOR) generally show a deviation of less than 5%, indicating a good match.

The water-cut show two areas of interest. One being of minor significance at 03-01-1998, while there is one point of major significance at 03-01-2001. These will be looked further into in the discussion.

The keyword "RATE" in the schedule file, control the injection rates for both water and gas continuously throughout the simulation. Subsequently, the values of both simulators should be identical. Looking at the graphs (Figure D.2), this seems to be true. However, scrutinizing the data reveals a small deviation in total water injection throughout the second half of the simulation history (Figure 4.6). Although the differences (Table 4.5) are insignificant, it implies that there are differences in individual injection wells.

The assumption of equal injection volumes is only valid if all other constraints are honored identically. For the Norne simulation, the only other constraint is a maximum pressure limit. The pressure restriction serves as a maximum

Table 4.5: End difference of the total cumulative injection. Eclipse as reference

Fluid type	Cumulative injection: Eclipse	Difference in cumulative injection values	Relative difference
	[SM3]	[SM3]	[%]
Water	$10.21 \cdot 10^7$	$4.97 \cdot 10^5$	0.487
Gas	$86.85 \cdot 10^9$	$10 \cdot 10^3$	0.00001

threshold. If the well requires more pressure to inject the set rate, the well must instead reduce the rate to honor pressure constraint. The difference in injection amount suggests that pressure restriction has been applied to one or more or of the wells, for different intervals of time. This would cause a difference in injection amount. Again the ensemble realization (Figure 4.7) show that this is a systematic error and the deviation does not shift significantly in the individual cases.

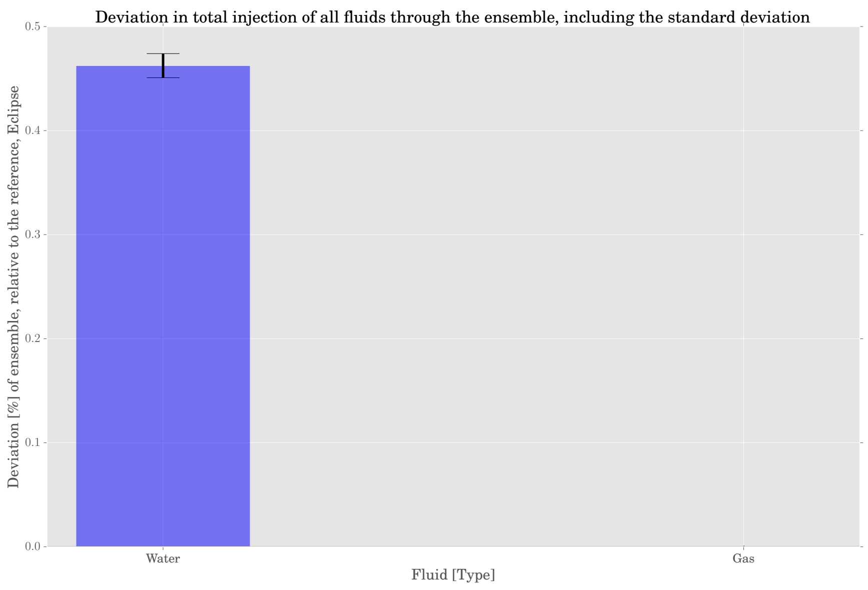


Figure 4.7: The figure illustrates the deviation in injection of water [blue] and gas [green] from the ensemble realization. The standard deviation is colored in black.

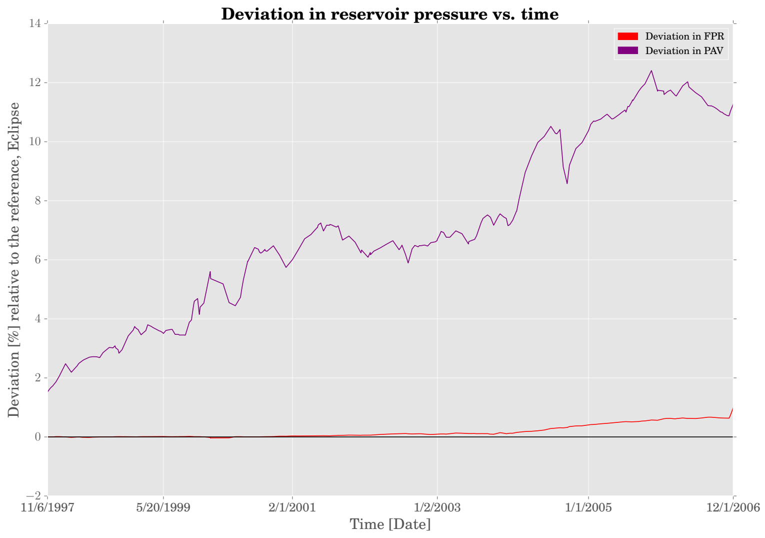


Figure 4.8: Deviation in Field Reservoir Pressure (FPR) [Purple] and Pressure Average (PAV) [Red], according to the .PRT file of Flow. Eclipse as reference.

The deviation in reservoir pressure between the two simulators (Figure 4.8) initially showed a significant difference. The summary file supplied by Flow gave a deviation in pressure of up to 12%. This difference would cause many issues in the reservoir. An in-depth look into the .PRT file (log file which contains events from the simulation) revealed that there was a discrepancy between the summary output and average pressure used by the Flow during simulation. The solution involved writing a script that searched through the .PRT file and recorded the average pressures throughout all report steps. This solution resulted in better consistency with the reference and showed a maximum deviation of 0.99% at the end of the simulation.

Figure D.4 and D.5 show the absolute difference cell-by-cell. A property filter with a magnitude of -1, -3, -5, and -10 Bar illustrate where the pressure cell-by-cell is less in Flow compared with the reference results. Subsequently, a property filter of 2, 3, 4, and 5 Bar of difference was applied to illustrate where the pressure in Flow is larger than the reference. These figures indicate that there is a small pressure problem within the G-segment of the reservoir, both above and under the *Not* formation. The largest deviation is located in the gas-layer and is considered significant.

Well Data

The field-level production data is one metric that determines the validity of the results. These are ultimately the parameters that will be the deciding factors in decisions for the development at a field-level, throughout the lifetime of the reservoir. However, a successful field development and operation of an asset comes down to the decisions made at well level. The iterative process of finding the optimal well placement is dependent on accurate simulation. Considerable attention should therefore go into analyzing all aspect of both production and injection wells, and the differences that may exist between Flow and the reference. After all, many a mickle makes a muckle.

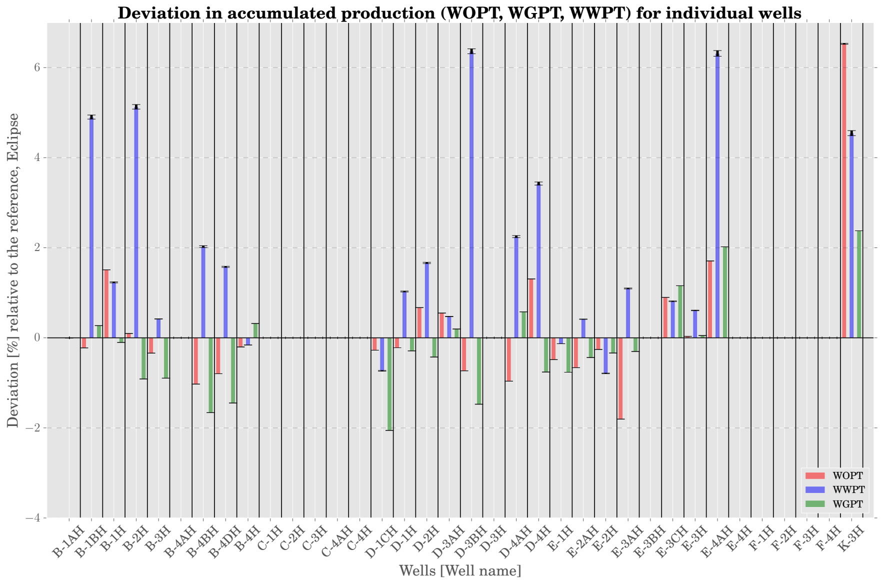


Figure 4.9: Deviation well gas production total [WGPT], well water production total [WWPT], and well oil production total [WOPT]. Black errors bar illustrates the standard deviation through the ensemble. Eclipse as reference.

The production data (Figure 4.9) reveals that there are some deviations in most production wells. Flow seems to produce higher amounts of water, with the largest deviations of 4.545% to 6.362% in deviation of cumulative production from wells: B-1BH, B-2H, D-3BH, E-4AH, and K-3H. The black error bars indicate that these deviations happen throughout the ensemble, due to this, they are systematic of nature and not due to abnormal behavior of this case. Regarding total oil production, there is only one well that produces deviation of minor significance: K-3H. Producer K-3H will be discussed in the discussion section of the case study.

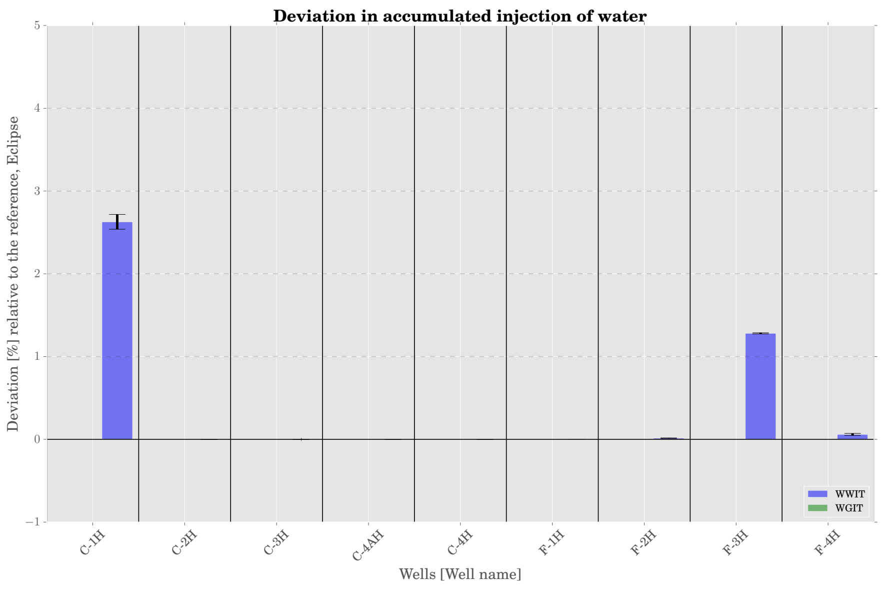


Figure 4.10: Deviation well gas production total [WGPT], well water production total [WWPT], and well oil production total [WOPT]. Black error bar illustrates the standard deviation of error in the ensemble. Eclipse as reference

Figure 4.10 illustrates the deviation concerning cumulative injection of gas and water. The cumulative gas injections remain consistent to the reference through all wells, giving a good match.

As previously mentioned, Figure 4.6, show a deviation in the cumulative injection of water. This difference originates from injectors: C-1H, F-2H, F-3H, and F-4H. Both simulators hit the pressure constraint and subsequently reduces the rate of injection to honor it. The difference may stem from the bottom-hole pressure profiles being different. If they are different, the wells will limit the rates with different amounts. Figure 4.10 show that the deviation is insignificant at only 2.63% and 1.3% for injectors C-1H and F-3H. Even though they play an insignificant role in the grand scheme of things, the wells show some unusual behavior and are subject to further investigation in the discussion section.

The bottom-hole pressure is one of the most, if not the most sensitive variable in any reservoir simulation. The bottom-hole pressure is directly dependent on multiple variables such as static properties, geometry, and rates. Figure 4.11 show the relative difference for each report step. The values for bottom-hole pressure show a maximum negative pressure difference of -3.93% in well E-4AH, and a maximum positive deviation of 1.8% in well D-1CH, compared with the reference. Overall, the mean deviation in all wells is insignificant. Outliers, which may skew calculations, are frequent in BHP data due to the fluctuating nature of the measurement, especially when rate specific keywords like 'RESV' and 'RATE' controls the wells according to historical values. These keywords abruptly change the rates, and thereby the bottom-hole pressure to either produce or inject the new target rate. Figure D.1 show the bottom-hole pressure graphs for wells B-1H, D-1H, E-3H, E-4AH and F-3H. On the left-hand-side, there is a distinct overlap between the BHP pressures. The right side of the figure shows the absolute difference in BHP pressure. It graphically shows that most wells are exposed to outliers, and not a general trend of deviation. One notable exception is E-4AH, which shows a tendency of being continuously lower than the reference, although at an insignificant level.

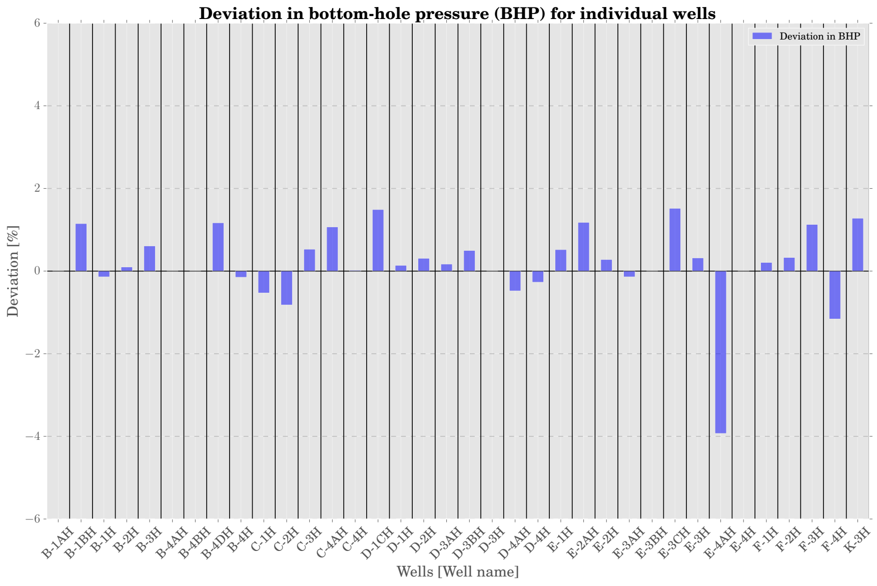


Figure 4.11: Mean of relative difference in bottom-hole pressure (BHP) [Blue] through all report steps

Saturation

The oil saturation (Figure E.1) show no real difference in saturation. There are a couple of individual cells showing some difference, although this difference is insignificant for the most part. Indicating, in line with the cumulative production Figure D.2, for a good match.

The gas saturation (Figure E.2) seem to be coherent between Eclipse and Flow. Through most of the layers, there are no significant differences. Except for an area, located in layer 17-18,(Red circle, Figure E.2, Layer K: 17-21) that show a difference between the two simulators. This patch located between injector C-1H and C-4AH, which both are water-alternating-gas (WAG) injectors show higher saturation of gas in the output from Flow compared with the reference. The saturation difference is of minor significance and has an insignificant impact on the output.

As for the gas saturation, the water saturation shows very similar behavior. The same area appears between injectors C-1H and C-4AH, (Red circle, Figure E.3, Layer K:17-21) showing reduced water saturation in Flow. As the oil saturation show no differences in this area (Figure E.1) it is likely directly related type of fluid injected by injector C-1H.

Static Properties

The static properties are mostly the same between Flow and the reference simulator, Eclipse. Except for one static property; transmissibility. Transmissibility is the parameter used to express the ability of a porous media to conduct fluids. There are two options for including this property into a simulation. The first possibility is to insert a set of transmissibilities to dataset explicitly. The other choice is to allow the simulator to calculate the transmissibility at the start of the simulation; however, Flow and Eclipse use two different methods to calculate this property. Norne uses the latter option, which causes a difference.

When considering the transmissibility, there are only one two areas that show some difference. In both Figure E.4 and Figure E.5 the red circle indicates the presence of a zone between WAG injectors C-1H and C-4AH, which show higher transmissibility in Flow for both X-direction and Y-direction, compared with the reference. This area located exactly where the difference in water and gas saturation appeared, suggesting that the deviation might be correlated.

When looking at the difference in transmissibility in the Z-direction, there is an area in the G-segment which show a significant deviation, which is the same segment as the cell-by-cell pressure profile illustrated differences (Figure D.5 and Figure D.4).

4.3.3 Performance

Assuming that the results are reliable, the next question of priority is the computational performance of the simulator. Computational time is expensive in terms of licenses fees, and resources - both personnel and equipment. Running ensembles is a common part of most workflows; it helps with the understanding of reservoir behavior, history matching and optimizing well placement. For this test, the ensemble was run in sequential order to idealize the situation regarding the available resources. Both Flow and Eclipse was run using both a single core and dual-core setup.

Run-time: Single-core

Figure 4.12 and Table 4.6 show the run-time distribution for all three setups. In terms of single-core performance, Eclipse has the upper hand. Flow has an increased mean-time utilization of 16%.

Table 4.6: Wall-time [s], single-core performance of both simulators. Licensing time is included. Eclipse time as reference

Simulator	Version	Mean run-time	Standard deviation	Relative difference
		[Seconds]	[Seconds]	[Fraction]
Eclipse	2016.1	746.535	9.872	1
Flow	2017.04.0	863.950	13.31	1.1573
Flow	2016.11.0	2030.950	50.48	2.720

These numbers are influenced by the time Schlumberger Simulation Launcher uses to fetch the Eclipse license. Including the licensing time was done to test the user experience, in addition, it is a drawback with Eclipse compared to Flow that directly relates to the development model. The licensing time is reflected in Table 4.7. It appears that the time spent fetching are different in Ubuntu

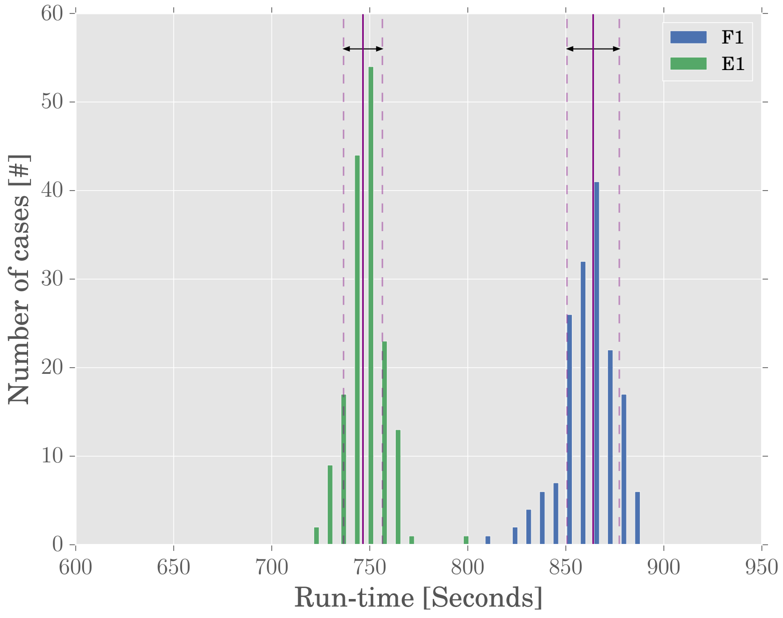


Figure 4.12: Run-time for single-core Flow 2017.04 [Blue] and Eclipse [Green], sequential solving. Double-headed arrow indicating the standard deviation

16.04_x86_64 and Windows 10_x86_64, both using the same VPN client; Anyconnect VPN.

Run-time: Dual-core Flow vs. Dual-core Eclipse

Flow recently introduced native support for Message-Passing Interface (MPI), which allows for two or more processors to communicate. This new feature enables Flow to use two cores simultaneously, to reduce computational time. Eclipse already has this feature its disposal, a test of this feature is, therefore, warranted. Table 4.8 and Figure 4.13 show the results. The results show that Flow has a better mean run-time compared with Eclipse, it suggest that Flow utilizes increased computational resources more efficiently than Eclipse. Another interesting observation is how Eclipse performs worse in a dual-core setup, compared with a single-core setup.

Table 4.7: Licensing time for Eclipse.

Simulator	Version	Mean run-time	Standard deviation
		[Seconds]	[Seconds]
Eclipse	Windows 10_x86_64	26.82	4.98
Eclipse	Ubuntu 16.04_x86_64	90.40	6.23

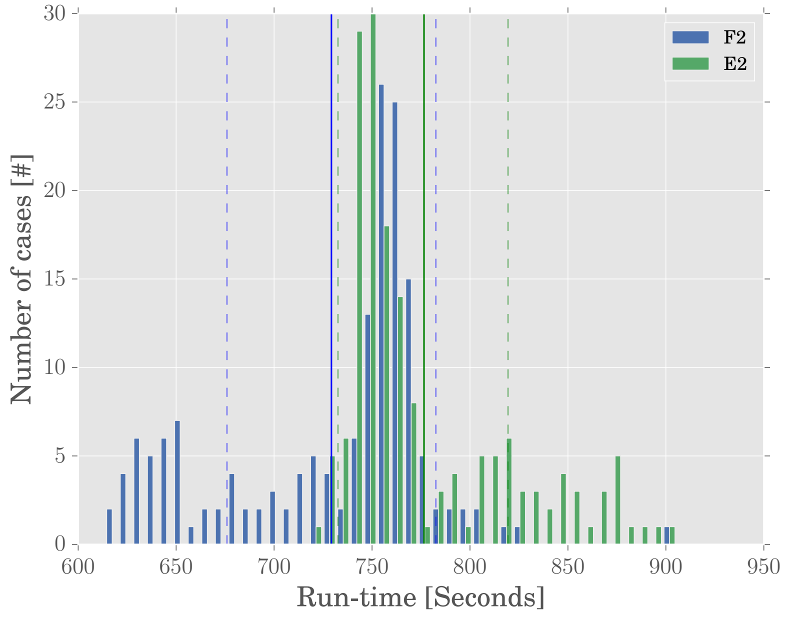


Figure 4.13: Run-time for dual-core Flow 2017.04 [Blue] and Eclipse [Green], sequential solving. Stipulated lines indicate standard deviation, and the continous line indicate mean run time (in their respective color)

Table 4.8: Walltime [s], single-core Eclipse performance and dual-core Flow performance. Licensing time is included. Eclipse time as reference

Simulator	Version	Mean run-time	Standard deviation	Relative difference
		[Seconds]	[Seconds]	[Fraction]
Eclipse	2016.1	776.0244	43.402	1
Flow	2017.04.0	729.27	53.30	0.939

4.4 Discussion on Validity and Performance for the Case Study

Through this section, there will be a discussion on the items that were mentioned in the result section and are due for further investigation. This section will also try to shed some light on the differences and why they might occur.

4.4.1 Oil Production

For the oil production, there were no large deviations. The field production has a good match with the reference. The only well showing a difference of minor significance is K-3H, this well starts production almost at the end of the simulation.

Table 4.9 examines K-3H in a larger context, it concludes with a declassification from minor significance to insignificant.

Table 4.9: Wells of interest in oil production, using data compiled to Figure 4.9

Well of interest	Deviation	Initial definition	Adjusted definition
K-3H	6.52 %	Minor significance	**Insignificant**
Examination: The well has been active for a total of 47 days and has produced a cumulative amount of 2 464 SM3 of oil. The difference in production is 151 SM3. Any deviation in the start will cause a significant error for relative change. If this was the case after a year or two, this would an error of minor significance			

4.4.2 Water Production and Injection

The field water production total shows a good match throughout the simulation history. The water-cut did, as previously mentioned, show two areas of interest. Table 4.10 show the context and adjusted definition of the water-cut. It reveals that seen in a larger context; the significance is minor. Both areas of interest happen when the water-cut is very low and before water-breakthrough occurs, causing the deviation to spike when exposed to minor change.

Table 4.10: Areas of interest in water-cut deviation graph. (Figure 4.5)

Date of interest	Deviation	Initial definition	Adjusted definition
3/1/1998	-7.84%	Minor significance	**Insignificant**
Examination: Water breakthrough has not happened in any well at this point. Therefore the water-cut is still minuscule (0.00069). Any absolute deviation in individual wells will have a significant impact on the relative deviation. Difference in field water production rate (FWPR) 0.071 SM3/day. No lasting impact on production values			
3/1/2001	12.72%	Major significance	**Minor significance**
Examination: Water breakthrough has not happened in any well at this point. Therefore the water-cut is minuscule (0.059240). Any absolute difference in the well will have a significant impact on the relative deviation. Difference in field water production rate (FWPR) 295.14 SM3/day. No lasting impact on production values.			

Although the deviation may seem significant, these numbers are insignificant as the total production from the well is low. Table 4.12 puts the deviation in a field-scale perspective. It shows that the wells that display the highest difference are the wells that produce the least amount. Due to this reason, over time, as production increases, the deviation would be in line with the other wells.

Figure 4.6 and 4.10, as previously mentioned, show some interesting phenomenons. The deviation is insignificant; however, these well illustrate some unusual behavior. Table 4.13 provides one possible explanation as to why differences in cumulative water injection happen in C-1H and F-3H occur.

In well F-3H there seems to be the presence of a keyword that is supported in Eclipse and not in Flow. In Eclipse the keyword stops the injection from F-3H, while in Flow ignores. This results in a spike of deviation for the bottom hole pressure (Figure D.1, F-3H) and cumulative water injection (Figure 4.10). The problem might either by the lack of support or a bug. Nevertheless, it illustrates one of the drawbacks of Flow, the number of keywords support by Flow is far less than Eclipse. After having reported this issue to the coordination team at OPM, they gave the impression that this issue was already being fixed.

Table 4.11: Deviation in well water production total for the wells with highest deviation

Origin of water production	Cumulative production of water (Eclipse)	Absolute deviation in cumulative production of water	Relative deviation
	[SM3]	[SM3]	
B-1BH	313 603	15 374	4.902%
B-2H	1 731 860	88 840	5.130%
D-3BH	177 718	11 307	6.362%
E-4AH	170 964	10 797	6.315%
K-3H	44	2	4.545%
Sum of the above	2 394 189	126 320	5.278%

Table 4.12: Deviation in well water production total for the wells with highest deviation compared to the total field water production

Origin of water production	Cumulative production of water	Absolute difference in cumulative production from wells in Table 4.11	Relative deviation in cumulative water resulting from wells in Table 4.11
	[SM3]	[SM3]	
Field water production	23 201 500	126 320	0.544%

4.4.3 Gas Production and Injection

Regarding gas production and gas injection, everything seems to behave the same way in both Flow and Eclipse. The cumulative injection of gas is identical, and the deviation in cumulative gas recovery is only -0.44%, in other words - Flow an insignificant amount less compared to the reference.

4.4.4 Pressure

The reservoir pressure profile of the simulation initially contained significant deviation. This deviation was not in line the cell-by-cell pressure difference, therefore,

Table 4.13: Wells of interest in terms of cumulative water injection. (Figure 4.10)

Well of interest Adjusted definition	Deviation	Initial definition	Examination
C-1H	2.63%	Insignificant	**Insignificant**
Examination: This well seems to be hitting the pressure constraint less in Flow compared with Eclipse, resulting in larger amount of water being injected into the reservoir. By examining the transmissibility in the area, there is evidence of the transmissibility being larger in the X and Y-direction, in Flow compared to Eclipse. This would result in lower pressure for the same injection rate.			
F-3H	1.28%	Insignificant	**Minor significance**
Examination: This well injects more water in Flow compared with Eclipse. Looking at the bottom-hole pressure Figure D.1 it appears to be a spike in pressure at the end, and the water injection seems to stop in Eclipse, but continue in Flow. A line in the schedule seems to stop the production in Eclipse.			

further investigation was initiated. It showed that the summary file, which is interpreted by ResInsight, contained the wrong information. Searching through the .PRT file (log file of the simulation) it became apparent that the PAV value, which is the simulations pressure average and the basis for calculation, utilized a different set of values. Scripting a program to extract these data and subsequently sort them by report steps revealed a better correlation. After consulting with the project lead, Alf Birger Rustad, he ensured that the problem was already fixed and was caused by a lack weighing towards the pore volume of the reservoir model.

4.4.5 Saturation and Static Properties

The saturation profiles show one particular area of interest. The location between well C-1H and C-4AH show increased oil saturation and reduced water saturation. Figure E.7 show the correlation between water and gas saturation through selected report steps. It shows that there are less water and more gas in the area of interest, throughout most report steps. Seen in context with the transmissibility and the bottom-hole pressure in the area (Figure 4.11, E.4, and E.5) it reveals that the transmissibility is calculated to be higher in Flow compared the reference, Eclipse. It also shows that the injector C-1H experiences less pressure for the equal rates. This behavior has no impact on the production, but it does allow injector C-1H to inject more water in Flow compared with the reference.

4.4.6 Run-time

The run-time show that there are differences regarding computational speed. The single-core performance (Figure 4.12) show that Eclipse is slightly faster. The wall-time (Table 4.6, human perceived time) show that Flow spends 15.73% more time, compared to Eclipse. Implying that computational time is getting on a competitive level, especially when taking into account that the previous version, Flow 2016.11, spent 272% more time compared to Eclipse. From 2016.11 version to 2017.04 a time reduction of almost 60% was achieved, indicating a rapid pace of innovation. These figures do include the time spent fetching a license from a centrally localized license server. Table 4.7 show the time spent both in Windows and Linux. Schlumberger Simulation Launcher (the application that fetches the license) spends less time in Windows 10 compared with Ubuntu 16.04. It is unclear why this is the case. However, it does highlight the fact that license acquisition does take a good portion of the time. Flow, on the other hand, does not need to fetch a license and thereby saves time. Although the licensing time is significant for this case, one should have in mind that as other assets with increased size and number of cells are run, the simulation time increases. While licensing time is fixed, the computational time varies. Therefore, in larger simulations, the licensing time will be increasingly marginalized, and raw computational time will be the factor of interest.

The dual-core Flow vs. Eclipse test (Table 4.8 and Figure 4.13) also illustrated that Flow has working MPI that increases computational speed when more resources are being utilized. It outperforms Eclipse in terms of computational speed, which essentially means that Flow is more scalable than the reference. In modern computers there are normally more than one core per case, Eclipse licenses which enables parallelized sequential solving costs more than the one-core license. This fact means cost reduction for companies looking to run ensembles, both in terms of computational time and license fees.

Although this test shows that the MPI works, it is a common understanding that parallel processing of the reservoir is better optimized when there are fewer report steps and larger cell count. This optimization is caused by a reduced time spent synchronizing the cores. Synchronizing the cores involves the cores to idle while waiting for the other cores to catch up to the current report step and communicate the results. This increase and complexity of a reservoir would mean less synchronization time and more time spent computing.

Overall these results show an impressive match between Flow and Eclipse. The results are supported by the base case and the ensemble which show little deviation both in the baseline scenario and within the ensemble. This benchmark test was performed with the use of history controlled mode; it reduces the number of potential errors compared to a predictive test. The simulations are tough and require a significant number of features to complete. A sensitivity study proved to be a real test for both simulators, and some interesting observations showed that the simulations were not identical, but the errors were systematical.

One of the drawbacks of Flow is the lack of support for niche keywords; this resulted in some unexpected behavior in the water injection (Figure 4.10, F-3H). Adapting cases to fit the simulator is a task that engineers do not prefer to do and may lead to uncertainty in the simulation. This stresses the need for Flow to increase development of niche keywords that currently are lacking.

4.4.7 Limitations

This benchmark study contains some limitations. As the tuning settings were discarded from the simulation, there were significant differences in the length of each time-step. This difference in length of the time-steps may cause truncation errors. An approach to eliminate or severely reduce the truncation error would be to force the simulators to take smaller time-steps by having shorter time intervals between report-steps.

Another limitation is the difference in static properties. The transmissibility is calculated differently, this could be eliminated by manually applying the transmissibility calculated by one simulator to the include file for both simulators. This modification would reduce the deviations associated with the transmissibility, and one could exclusively focus on evaluate the fluid behavior from the simulation.

4.4.8 Summary and Conclusion of Benchmark Study

Considering the run-time, Eclipse run faster in single-core environment. The reverse is true for the dual-core test, it is thereby illustrated that given more resources Flow is competitive and can be a good alternative to Eclipse. This benchmark study, therefore, shows that an open-source reservoir simulator can perform competitively without sacrificing reliability.

Norne is a complex reservoir with lots of features. The benchmark study has illustrated that the overall level of accuracy of Flow, under history-controlled mode, is very closely matched with the reference simulator, Eclipse. There are a few areas of interest, showing a minor level of significance, especially under the topic of water production and injection. These differences have an insignificant impact on overall production values and are replicated throughout the ensemble. The ensemble shows little deviation even when the parameters are changed for both simulators. This indicates that the deviation from Eclipse are systematic and not by chance. The conclusion, in line with the results, is that an impressive match is achieved by Flow compared to Eclipse.

s

Chapter 5

Discussion

The purpose of this paper was to explore how the deployment of Integrated Operations (IO) has affected the workflows and communication for engineers in the oil and gas industry. How an increasing level of innovation, especially within the domain of software development, can further improve and fundamentally change the way engineers work with industrial scale projects. Given the context of Integrated Operations, it is clear that most initiatives within this domain start with an idea of change. Change for how engineers collaborate across disciplines for better decisions, change in engineering workflows to improve efficiency, change the governance models to better stimulate for cross-domain learning, and change the mentality of an organization to embrace contracts which improve the flow of information between service companies and operators.

To be able to discuss Integrated Operations it was necessary to give some insight into how the industry itself views the deployment of IO. Through several interviews, the subjects were asked the question of how do you interpret the term Integrated Operations. The answers were similar, it contained the official definition, although when getting more into the specifics the answer dependent on the career and educational background of the interviewee. When asking software engineers, they would refer to digitalization, live data feeds, and innovations within the software service industry. When petroleum specific discipline engineers were asked, they would talk about the co-localization and improvements within video-conferencing, planned maintenance, and new technology that allow for remote assistance for field operations. It highlights the fact that there are multiple interpretations of Integrated Operations. The overarching theme is a design philosophy which stimulates collaboration between disciplines, increased automatization and the transition to new, more efficient workflows.

Digitalization is the enabling dimension that allows for the deployment of IO. Collaborative environments are used more than ever to maximize production and reduce costs. Off-shore and On-shore facilities are now receiving the same live data from all sensors, both during drilling operations and production. In theory, on-shore personnel could remotely control the production facility in its entirety. Virtual collaborative environments maintain a level of communication that would not be feasible in 2005. In a visit to AkerBPs control room, which has the shared responsibility for operations at Ivar Aasen, they expressed some of the pillars the control room was built according to; one of which was trust between coworkers. Through the development of Ivar Aasen, both offshore and on-shore crew worked together to create personal relationships that would aid them in communication and also know each others strengths and weaknesses. Interviewees expressed that operation would be impossible if this crucial step were not taken already in the development phase. These experiences show that the development of new work processes has come far. Engineers are, in fact, working better together, through co-localization, reduced amount of travel and more collaboration between different engineering domains from various geographical locations.

In 2005 Norwegian Oil industry Association attempted to predict the deployment of Integrated Operations. The report stated that within the span of a decade the second generation of IO would be fully deployed. Through several interviews with employees of AkerBP and Statoil, they have highlighted the fact that Integrated Operations have come far, but not to the level that was foresighted in the report by Norwegian Oil industry Association (2005). At the time of the report, a capability approach and an open innovation environment was never considered. This approach has been developed through work like Henderson, Heps, Mydland, (2013). The development considers Integrated Operations as an information ecology, rather than, solely as an organizational tool. An information ecology, as presented in chapter 2, directly relates digitalization as an integral part IO. This inclusion shifts the dynamics of IO and improves the technological understanding.

Through the deployment of IO, it is important to differentiate between the dimensions of the philosophy itself. Some of these dimensions are developed further than other, and this causes improved efficiency combined with certain drawbacks. For realizing the potential within Integrated Operation, it is essential that all dimensions are developed simultaneously and adapted to fit each other.

One of the main assumptions, as mentioned in chapter 2, NOAs 2005 report was that people, workflow, and organization were the remaining factors, which were delaying the deployment of the first and second generation. This assumption is inherently flawed, as it presented the technology as ready. Technology itself is a huge term; it includes all the sensory equipment, production facilities, and software that control the new, increasingly, automated field developments.

Closed-source development and licenses present the software vendors with the easiest way of turning a profit. Protecting the software by licenses is a predictable and safer than the alternatives, by reducing the risk of competition within the set domain. But, by protecting itself from competition, the vendor is turning the simulator into a black-box. The customer is presented with the simulator as an external entity, and the user interfaces only accept input of data and interpretation of the results. Valuable information is thereby lost, as there is no way of gaining further insight to the simulator. The option has so far been to use specific time-consuming black-box tests. Eclipse is well documented through both its manuals and technical descriptions. The drawback of this situation is that it only provides the answers to what the developer in-house at Schlumberger believe is useful, as a customer one is not able to explore beyond this and to test possible modifications.

Open-source development and licenses are found at the other end of the software scale; they offer increased freedom for the end-user. When software is distributed using these types of licenses, customers can, at their discretion feel free to explore and modify individual parts of the software. They can use white-box testing techniques to discover quirks and other abnormal behavior. If a problem is identified, the user can report the problem to coordination team, or better yet, offer their solutions for increased response time. This increase in freedom would allow anyone to familiarize themselves with the inner workings of the simulator, to further enhance their understanding by observing the softwares reasoning, limitations, and uncertainty.

As the upstream industry is considered an information sensitive one, security is a serious concern. Some common beliefs are that open-source software is less secure than their proprietary counterpart. In a study, it was revealed that there is no significant security difference between an open-source and closed-source development model (Schryen, 2011). Netscape was in its original state, a closed-source internet browser. It was designed with efficiency and security in mind but found itself in stiff competition. In 1998 Netscape announced that they would make the source-code available, and gave the project lead to Mozilla.org (Mockus, 2002). Skipping ahead to present day; Mozilla Firefox (the derivative product) has a remarkable market share and is widely regarded as one of the most secure web browsers. (Tip Top Security, 2014). It illustrates that by altering the fundamental terms of the availability of the code, you can develop at reduced costs and higher flexibility without sacrificing security.

In a proprietary model, when building software, there is an ever-present conflict of interest. Customers will always request features and adoption of niche technologies, sometimes more features are better, under other circumstances they might act more like a curse. If an executive decision is made to pursue customer wishes, often the software will become increasingly involved, and as a result, increase the difficulty of maintaining and continuously improving the core functionality. This over-complication might already be the case with Petrel. The platform contains hundreds of niche applications, requested by hundreds of clients. To keep up with the development of these applications Schlumberger must commit resources to maintain the different features, instead of focusing on what made the platform successful in the first place. This increase in complexity of the product is called scope creep (source management book), and this, if inefficiently managed will lead to excessive costs for development and in the end increased costs for the end-user. In an open-source development environment, the problem of scope drift is often avoided by the nature of how open-source projects are structured. Open-source software is usually not built from scratch; instead, they build on top of each other. An excellent example is how Flow uses the core functionality of an independent project called DUNE (OPM, 2017). DUNE is a modular toolbox for solving partial differential equations (PDEs) with grid-based methods (DUNE, 2017). Flow applies this frame to perform the calculations on the fluid behaviour in the reservoir grid. DUNE itself is developed through another open-source initiative, which focuses on streamlining its essential features. This creates dependencies, every incremental innovation at a lower level of dependency benefits the products that build on top of these projects. This module-based structure allows every open-source project to focus on the core functionality, and also gain from the accomplishments of other projects.

The key to success for any open-source software is the ability to create a community surrounding the software. This community supports the original project and increases the capacity of projects like OPM to implement new functionality, identify and fix bugs. The complexity and the development of additional features in an open-source project grow with the number of peripheral developers and projects that may or may not be in an initiative of the original project.

The structure of such a community should consist of committed operators, universities, and research institutions. Open-source projects have in the past shown that if the such a community managed properly, it can implement additional features, improve existing framework and fix problems in a pace far exceeding that of the proprietary development model, due to the sheer number of people involved. The challenge is to create this community and engage industry partners to an extent where they are willing to contribute towards further development. This problem could be overcome by verifying the simulator against a reference simulator and prove the viability of the alternative.

The benchmark study illustrated that Flow can provide accurate and consistent results compared to the reference simulator, Eclipse. Most deviations were insignificant, and had little impact on total production or individual well data. The benchmark study also revealed some of the drawbacks of Flow; it does not support the same number of keywords that Eclipse do, and therefore showed some abnormal behaviour, particularly in one injector. Nevertheless, the match between Eclipse and Flow was observed to be good.

The Norne benchmark study also tested the computational performance of the two simulators. When the ensemble was run in a single-core environment, the results showed that Eclipse slightly outperformed Flow. In a modern computational environment, there are usually more than one core available. The benchmark test for dual-core performance showed that Flow, contrary to the single-core test, slightly outperformed Eclipse. These results indicate that when more resources are available Flow is more efficient than Eclipse at utilizing the additional resources. It is impossible to investigate why this might be the case, as Eclipse is distributed with a proprietary license it offers no possibility of investigating beyond the technical description.

The results for a dual-core environment is particularly intriguing as they indicate slightly faster run-times for Flow compared to Eclipse. They show that the scalability of Flow is excellent. Running ensembles on multiple cores at a time is common. By running Flow instead of Eclipse has the potential to save operators both resources concerning computational time and license costs. This scalability would be a direct benefit of using Flow and is a persuasive argument for operators looking to reduce expenses in a long-term perspective.

The case study was conducted in history-controlled mode, which means that it did not predict what would come. To verify whether Flow is indeed a valid competitor to Eclipse, a verification study, therefore, needs to be conducted on predictive simulation. Nevertheless, the results were impressive and gave an indication of the alternative that can be expected in the near future.

Eclipse dominates the current market of reservoir simulators. A situation which benefits nobody but the monopolist. As in economics, the monopolist is free to maximize profit rather than delivering the best possible product. If operators partner with OPM or another open-source initiative and support the development of simulators such as Flow, it is feasible to force a shift in the market and increasing competition. This shift in market structure has the potential of giving benefits at both ends; it forces existing suppliers to innovate, reduce prices, and differentiate themselves from the competition. At the other end, it allows for a competitive solution in the form of OPM to allow for innovation in the shape of a co-development open-source model. It is only when facing competition that the monopolist shifts focus from maximizing profit to offering the best possible product.

Through the last three years, the oil and gas industry has undergone significant changes. Renewed attention to improving efficiency has surged. It is a sense of urgency by operators invest in solutions and apply cost-sharing partnerships to reduce costs in the long term. Even if it means revealing some of the trade secrets that are currently being kept in-house. OPM is an ideal candidate for the cost-sharing model. Operators could partner up with OPM in exchange for the development and implementation of keywords which are applied to their already existing models.

The openness offered by open-source software and the ability to modify the simulator to fit the needs of the customers are some of the keys to increase efficiency in everyday workflows. Operators no longer see themselves only in the role of a client, rather, they are continuously developing and maintaining their in-house scripts, software, and third-party solutions. Tighter integration between the simulator and third-party solutions could increase computational performance and cost fewer resources. Combined with the transparency that allows for close inspection of the simulator itself will allow for engineers to question individual parts, it will potentially raise the level of autonomy in workflows like history-matching.

Even if a launch of OPM is unsuccessful, it provides opportunities for the industry. The nature of open-source allows other initiatives to rise in the wake of OPM. This ensures that there will always be a threat to the status quo.

As pointed out by several interviewees user-interface and user-friendliness are topics that have to be addressed. Currently, Petrel and Eclipses combined offers an ease of use that is unparalleled by any other subsurface software platforms. Flow, on the other hand, requires a pre-existing input deck to run. OPM offers no alternative regarding making a reservoir model. To be a viable replacement for Petrel and Eclipses other open-source projects have to be involved to produce one such way of input. The co-development structure is both the biggest benefit and greatest challenge for an open-source project. Making a sustainable platform and business model is vital for any business, regardless of licensing scheme. There are multiple approaches both by regards to providing the right services and targeting the right customers.

One such approach to offer supplementary services for simulations of large ensembles. Switching from a replacement to a supplement would in the immediate future enable operators with the ability to simulate an increased number of cases for a reduced price. As a complement to the existing simulator, the results could be verified by Eclipse before delivery to satisfy the requirements of license partners. Flow would offer operators a unique way of ensuring their ability to operate, and thereby increasing operators leverage towards service companies. The assured ability to operate would be a counter-measure to typical lock-in situations.

Chapter 6

Summary and Conclusion

This section contains the summary, conclusion and recommendations for further work.

6.1 Summary and conclusion

Through this thesis, several topics have been investigated and explained. Integrated Operations is a complex term, and as presented in chapter 2, it works in multiple dimensions to increase automation, digitalization and cross-boundary knowledge. Technology is both the enabling factor and one of the the major areas of challenge. It is therefore essential to adapt to a new technological age, and find solutions that might not have been foreseen.

Open-source vs. proprietary software development is one such area. The traditional practice is to buy services and software from companies that specialize in the given domain. It provides with the essential tools for communication and simulation. This model is old, but has proved its efficiency through decades. All the while, software development has come a long way from its infancy, it has been proven through projects, such as the UNIX core, that large scales endeavors can be produced, given the correct leadership. The UNIX core, which now is an essential piece of any large scale company, has proven to be reliable and adaptable; giving birth to projects like Red Hat, Ubuntu and Fedora.

Operators no longer only play the role of the customer, as the most used methods to input, extract and process data from simulations are scripts and third-party programs like the Ensamble based Reservoir Tool (ERT, 2017). These scripts and tools are usually based on third-party software and work in an inefficient manner, as they have to treat the simulation itself as an external entity. This gap of adaptability of the scripts surrounding and the simulation itself discourages autonomy and streamlined work flows, due to the lack of crucial insight and the ability to modify. Through open-source projects, like the Open Porous Media Initiative, the operator would have the ability to modify the simulator to fit the exact needs, with a level of transparency that would be required to further automate processes like automated history-matching.

Following the results from the Norne benchmark study, it is evident OPM is well on its way to develop a competitive reservoir simulator - on par with the reference, Eclipse. It is a great starting step, but as many of the interviewees pointed out; the engineers utilizing the software needs an incentive. In short term, it would have to either be speed or accuracy, the adaptability would only apply for those companies that have engineers or task forces that are able to actively engage in the project, and modify the simulator to the needs of specific assets. Majority therefore comes down to the educational background and experience of the individual engineer, and the willingness to see a long term perspective.

Introducing an open-source alternative to the market of reservoir simulators is important, as the current monopoly benefits no operator. One possible approach for OPM, could be to act as a supplement to Eclipse, or other reservoir simulator. It does not have to be an immediate replacement, but it would reduce the license fees associated with the realization of large ensembles, in conjunction with meeting the partners requirements by verifying models in Eclipse, or any other proprietary alternative. With the current focus on cost saving, the ability to partner up with multiple operators would make this a powerful incentive, both for OPM as a developer and any potential operator looking to reduce costs. It is a difficult process to get through, but, if done correctly would offer operators with more customization at a lower price point.

6.2 Recommendations for Further Work

In the spirit of transparency, all data that was applied to this thesis have been uploaded to DropBox (`https://www.dropbox.com`) and is available for those who would like to investigate further, check the results or would like to view the output manually.

Exact url can be obtained by a request to either: `bragesk@stud.ntnu.no` or `brage_sk@hotmail.com`

What became apparent during the thesis was that verifying a simulator is an enormous task. The more you learn, the more you would like to investigate. The hardest part of the thesis was to limit the scope and leave some tasks to be completed at a later stage, these include and are not limited to:

- Short-term
    1. Benchmark Flow and Eclipse forcing the simulator to take similar time-steps, this would significantly reduce the amount of truncation error associated with varying time steps.
    2. Study how Flow and Eclipse handles predictive behavior, and compare the results.
- Medium-term
    1. Investigate what stakeholders are interested in partnering with OPM
    2. Explore which projects are currently utilizing OPM as a tool
    3. Investigate the current list of words that are supported by Eclipse and not by Flow
- Long-term
    1. Investigate what competencies are needed by reservoir engineers to modify the simulator
    2. Investigate the need for operators to keep trade secrets and their willingness for cost-sharing in a low-cost environment
    3. Explore different business models for the Open Porous Media Initiative and introduce OPM as a low cost alternative to universities without partnerships with e.g. Schlumberger

Appendix A

Acronyms

This appendix contains some of the commonly used abbreviations throughout the thesis.

IO Integrated Operations

OPM Open Porous Media Initiative

CWE Collaborative Work Environments

FWIT Field Water Injection Total

FGIT Field Gas Injection Total

FWIR Field Water Injection Rate

FGIR Field Gas Injection Rate

FOPT Field Oil Production Total

FLPT Field Liquid Production Total

FGPT Field Gas Production Total

FWPT Field Water Production Total

FOPR Field Oil Production

FLPR Field Liquid Production Rate

FGPR Field Gas Production Rate

FWPR	Field Water Production Rate
FWCT	Field Water Cut Total
FGOR	Field Gas Oil Ratio
FPR	Field Pressure Reservoir
PAV	Pressure Average
WBHP	Well Bottom Hole Pressure
WOPT	Well Oil Production Total
WGPT	Well Gas Production Total
WWPT	Well Water Production Total
WOPR	Well Oil Production Rate
WLPR	Well Liquid Production Rate
WGPR	Well Gas Production Rate
WWIT	Well Water Injection Total
WGIT	Well Gas Injection Total
WWIR	Well Water Injection Rate
WGIR	Well Gas Injection Rate

Appendix B

Formation Description

Table B.1: Geological description of the reservoir wide formations in the Norne reservoir. Lithology and depositional environment is based on Dalland et. al. (Eds.), 1988.

Formation name	Property	Description
Garn	Lithology	It consists of medium to coarse-grained, moderately to well-sorted sandstones. Mica-rich zones are present. The sandstone is occasionally carbonate-cemented.
	Depositional environment	It may represent progradations of braided delta lobes. Delta top and delta front facies with active fluvial and wave-influenced processes are recognized.
Not	Lithology	Claystones with micronodular pyrite coarsen upwards into bioturbated fine-grained sandstones which are locally micra-rich and carbonate cemented.
	Depositional environment	The basal part of the formation reflects a semi-regional transgression which led to the development of lagoons or sheltered bays. The upper part of the unit consists of prograding deltaic or coastal front sediments.
Ile	Lithology	Fine to medium and occasionally coarse grained sandstones with varying sorting are interbedded with thinly laminated siltstones and shales. Mica-rich intervals are common. Thin carbonate-cemented stringers occur, particularly in the lower parts of the unit.
	Depositional environment	The basal part of the formation reflects a semi-regional transgression which led to the development of lagoons or sheltered bays. The upper part of the unit consists of prograding deltaic or coastal front sediments.

Table B.2: Continuation of Table B.1, description of the reservoir wide formations in the Norne reservoir. Lithology and depositional environment is based on Dalland et. al. (Eds.), 1988.

Formation name	Property	Description
Tofte	Lithology	The Tofte Formation consists of moderately to poorly sorted coarse-grained sandstones which often show large-scale cross bedding. In the type section the quartz content is generally higher than 90%, although the sediment is texturally immature. Bioturbation occurs throughout the cored intervals, especially in zones of very poor sorting and high clay content.
	Depositional environment	The sandstones were deposited by eastwards prograding fan deltas which reflect tectonic uplift to the west.
Tilje	Lithology	Very fine to coarse-grained sandstones are interbedded with shales and siltstones. The sandstones are commonly moderately sorted with a high clay content and most beds are bioturbated. Shale clasts and coaly plant remains are common. Pure shale beds are rare; most of the finer grained interbeds are silty or sandy.
	Depositional environment	Nearshore marine to intertidal environments are typical of the formation. Subcrops near the coast (Bugge et al. 1984) indicate a gradual transition to continental environments eastwards.

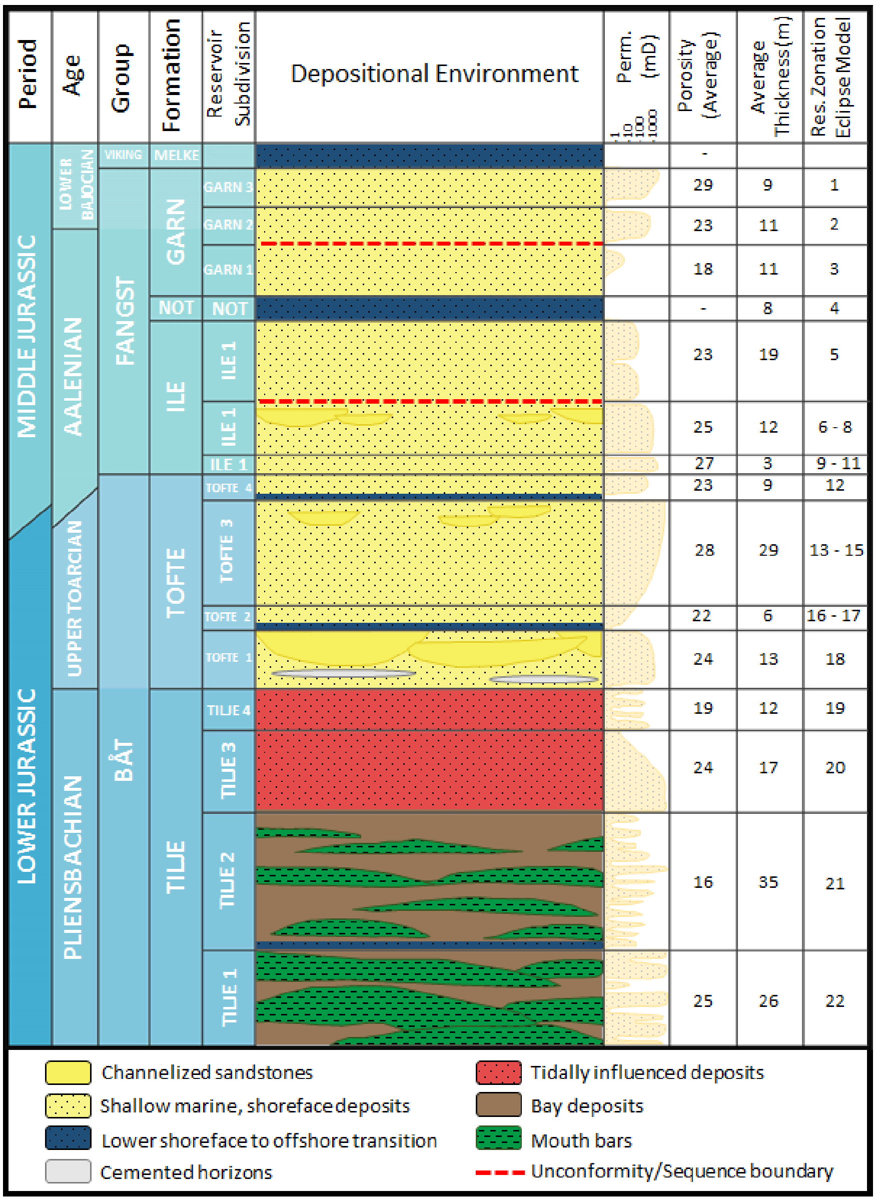


Figure B.1: Stratigraphic Column of the Norne reservoir. (Modified after Statoil, 2001)

Appendix C

Ensemble

Table C.1: Initial values of MULTFLT, as included with the Norne reservoir model

Name of fault	Initial value	Name of fault	Initial value
’E_01’	0.01	’C_21_Ti’	0.001
’E_01_F3’	0.01	’C_22’	0.001
’DE_1’	3.9	’C_23’	0.1
’DE_1_LTo’	0.01	’C_24’	0.1
’DE_B3’	0.00075	’C_25’	0.1
’DE_2’	0.015	’C_26’	0.1
’DE_0’	20	’C_26N’	0.001
’BC’	0.1	’C_27’	0.05
’CD’	0.1	’C_28’	1.0
’CD_To’	0.01	’C_29’	0.1
’CD_B3’	0.1	’DI’	0.1
’CD_0’	1	’DI_S’	0.1
’CD_1’	0.1	’D_05’	0.01
’C_01’	0.01	’EF’	1.0
’C_01_Ti’	0.01	’GH’	1.0
’C_08’	0.01	’G_01’	0.05
’C_08_Ile’	0.1	’G_02’	0.05
’C_08_S’	0.01	’G_03’	1
’C_08_Ti’	1	’G_05’	0.5
’C_08_S_Ti’	1	’G_07’	0.05
’C_09’	0.1	’G_08’	0.05
’C_02’	0.01	’G_09’	0.05
’C_04’	0.05	’G_13’	0.05
’C_05’	0.1	’H_03’	1.0
’C_06’	0.1	’IH’	1.0
’C_10’	0.01	’m_east’	1.0
’C_12’	0.1	’m_east_2’	1.0
’C_20’	0.5	’m_north’	1.0
’C_20_LTo’	0.5	’m_northe’	1.0
’C_21’	0.001	’m_west’	1.0

Table C.2: Initial values of MULTPLY, as included with the Norne reservoir model

Name of layer	Property	Initial value
Garn 3	PERMZ	0.2
Garn 2	PERMZ	0.04
Garn 1	PERMZ	0.25
Not	PERMZ	0.0
Ile 2.2	PERMZ	0.13
Ile 2.1.3	PERMZ	0.13
Ile 2.1.2	PERMZ	0.13
Ile 2.1.1	PERMZ	0.13
Ile 1.3	PERMZ	0.09
Ile 1.2	PERMZ	0.07
Ile 1.1	PERMZ	0.019
Tofte 2.2	PERMZ	0.13
Tofte 2.1.3	PERMZ	0.64
Tofte 2.1.2	PERMZ	0.64
Tofte 2.1.1	PERMZ	0.64
Tofte 1.2.2	PERMZ	0.64
Tofte 1.2.1	PERMZ	0.64
Tofte 1.1	PERMZ	0.016
Tilje 4	PERMZ	0.004
Tilje 3	PERMZ	0.004
Tilje 2	PERMZ	1
Tilje 1	PERMZ	1

Table C.3: Initial values and modified values of MULTFLT, as included with the Norne reservoir model

Name of fault	Initial value	Lower value	Upper value	Name of fault	Initial value	Lower value	Upper value
'E_01'	0.01	0.005	0.002	'C_21_Ti'	0.001	0.0005	0.002
'E_01_F3'	0.01	0.005	0.02	'C_22'	0.001	0.0005	0.002
'DE_1'	3.9	1.95	7.8	'C_23'	0.1	0.05	0.2
'DE_1_LTo'	0.01	0.005	0.02	'C_24'	0.1	0.05	0.2
'DE_B3'	0.00075	0.000375	0.0015	'C_25'	0.1	0.05	0.2
'DE_2'	0.015	0.0075	0.03	'C_26'	0.1	0.05	0.2
'DE_0'	20.0	10.0	40.0	'C_26N'	0.001	0.05	0.2
'BC'	0.1	0.05	2.0	'C_27'	0.05	0.025	0.1
'CD'	0.1	0.05	0.2	'C_28'	1.0	0.5	2.0
'CD_To'	0.01	0.005	0.02	'C_29'	0.1	0.05	0.2
'CD_B3'	0.1	0.05	0.2	'DI'	0.1	0.05	0.2
'CD_0'	1.0	0.5	2.0	'DI_S'	0.1	0.05	0.2
'CD_1'	0.1	0.05	0.2	'D_05'	0.01	0.005	0.02
'C_01'	0.01	0.005	0.02	'EF'	1.0	0.5	2.0
'C_01_Ti'	0.01	0.005	0.02	'GH'	1.0	0.5	2.0
'C_08'	0.01	0.005	0.02	'G_01'	0.05	0.0025	0.1
'C_08_Ile'	0.1	0.05	0.2	'G_02'	0.05	0.0025	0.1
'C_08_S'	0.01	0.005	0.02	'G_03'	1.0	0.5	2.0
'C_08_Ti'	1.0	0.5	2.0	'G_05'	0.5	0.25	1.0
'C_08_S_Ti'	1.0	0.5	2.0	'G_07'	0.05	0.025	0.1
'C_09'	0.1	0.05	0.2	'G_08'	0.05	0.025	0.1
'C_02'	0.01	0.005	0.02	'G_09'	0.05	0.025	0.1
'C_04'	0.05	0.025	0.1	'G_13'	0.05	0.025	0.1
'C_05'	0.1	0.05	0.2	'H_03'	1.0	0.5	2.0
'C_06'	0.1	0.05	0.2	'IH'	1.0	0.5	2.0
'C_10'	0.01	0.005	0.02	'm_east'	1.0	0.5	2.0
'C_12'	0.1	0.05	0.2	'm_east_2'	1.0	0.5	2.0
'C_20'	0.5	0.25	1.0	'm_north'	1.0	0.5	2.0
'C_20_LTo'	0.5	0.25	1.00	'm_northe'	1.0	0.5	2.0
'C_21'	0.001	0.0005	0.002	'm_west'	1.0	0.5	2.0

Table C.4: Initial values of MULTPLY, as included with the Norne reservoir model

Name of layer	Property	Initial value	Lower value	Upper value
Garn 3	PERMZ	0.2	0.1	0.4
Garn 2	PERMZ	0.04	0.02	0.08
Garn 1	PERMZ	0.25	0.125	0.5
Not	PERMZ	0.0	0.0	0.0
Ile 2.2	PERMZ	0.13	0.065	0.26
Ile 2.1.3	PERMZ	0.13	0.065	0.26
Ile 2.1.2	PERMZ	0.13	0.065	0.26
Ile 2.1.1	PERMZ	0.13	0.065	0.26
Ile 1.3	PERMZ	0.09	0.045	0.18
Ile 1.2	PERMZ	0.07	0.035	0.14
Ile 1.1	PERMZ	0.019	0.095	0.38
Tofte 2.2	PERMZ	0.13	0.065	0.026
Tofte 2.1.3	PERMZ	0.64	0.32	1.28
Tofte 2.1.2	PERMZ	0.64	0.32	1.28
Tofte 2.1.1	PERMZ	0.64	0.32	1.28
Tofte 1.2.2	PERMZ	0.64	0.32	1.28
Tofte 1.2.1	PERMZ	0.64	0.32	1.28
Tofte 1.1	PERMZ	0.016	.008	0.032
Tilje 4	PERMZ	0.004	0.002	0.008
Tilje 3	PERMZ	0.004	0.002	0.008
Tilje 2	PERMZ	1.0	0.5	2.0
Tilje 1	PERMZ	1.0	0.5	2.0

Appendix D

Production data and pressure profile

This appendix contains all relevant production graphs. In order to be able to present in an orderly fashion it was necessary to include much of the unprocessed, raw data, here. This is for further review by others that may want to continue the research or investigate in detail

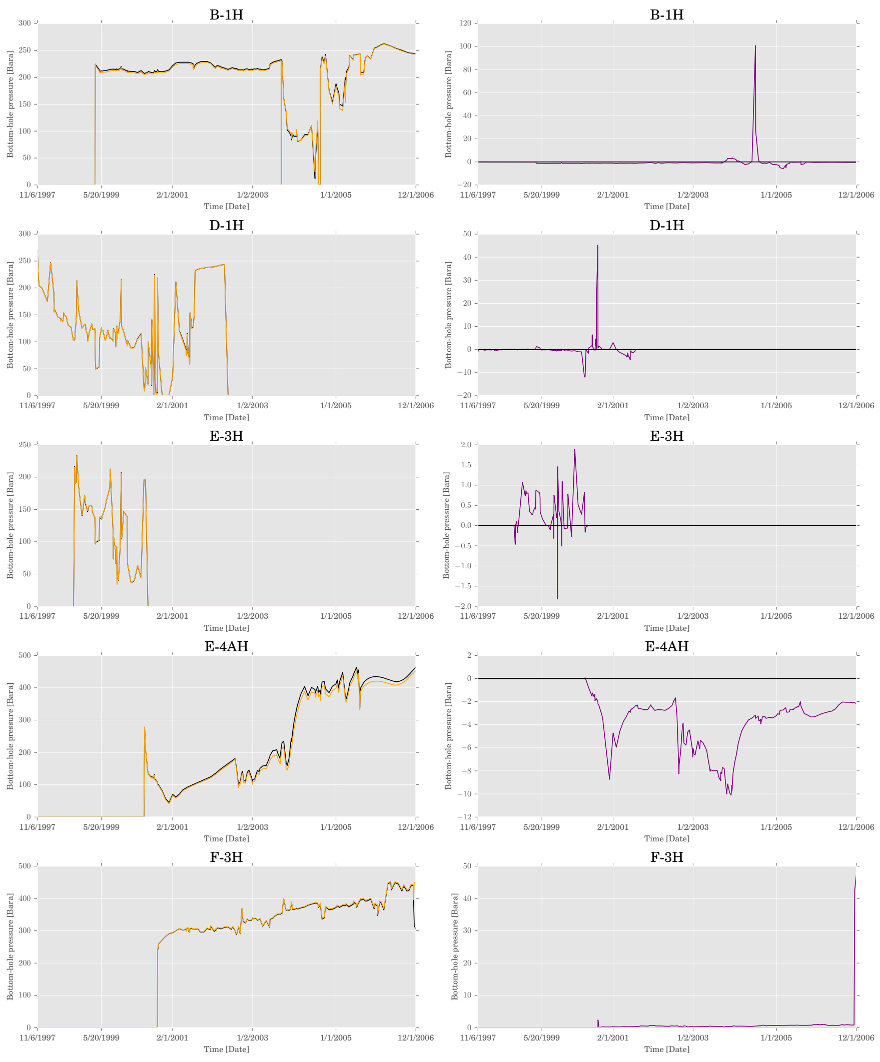


Figure D.1: Bottom-hole pressure data for wells that are most biased towards outliers (B-1H, D-1H, E-3H, E-4AH, F-3H)

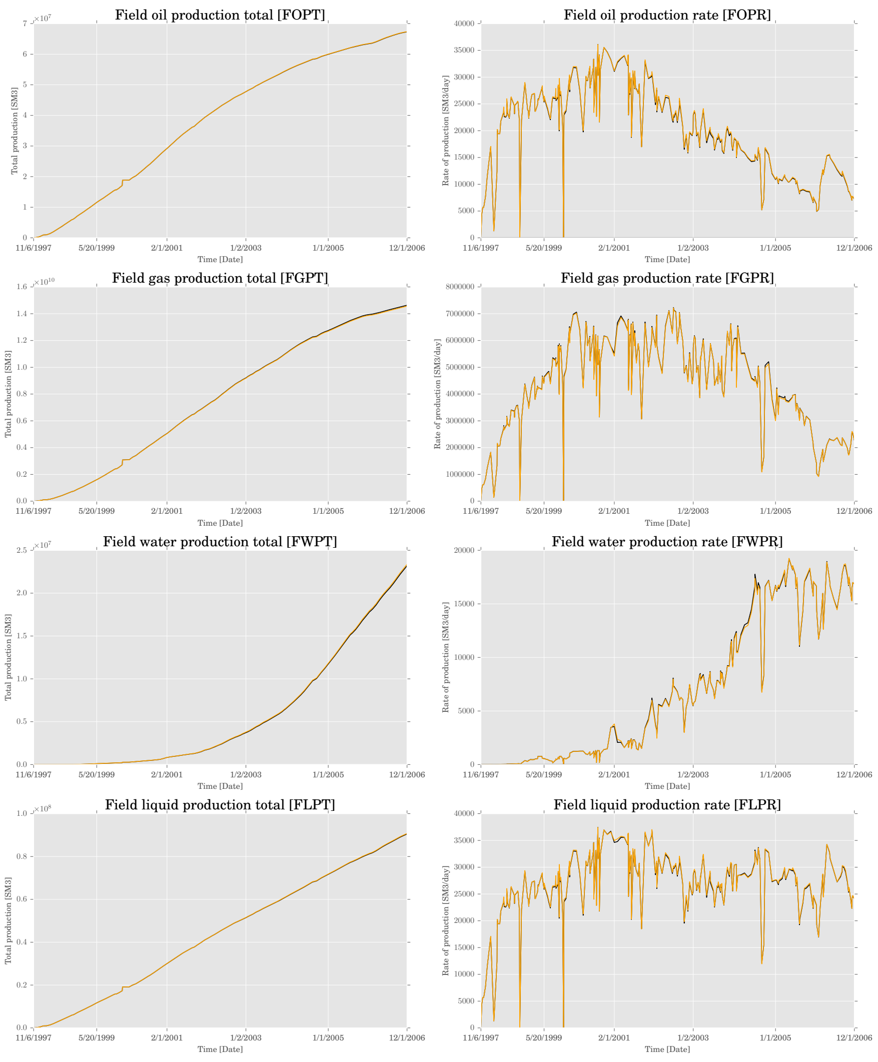


Figure D.2: Total production and rates for OIL, GAS, WATER and LIQUID

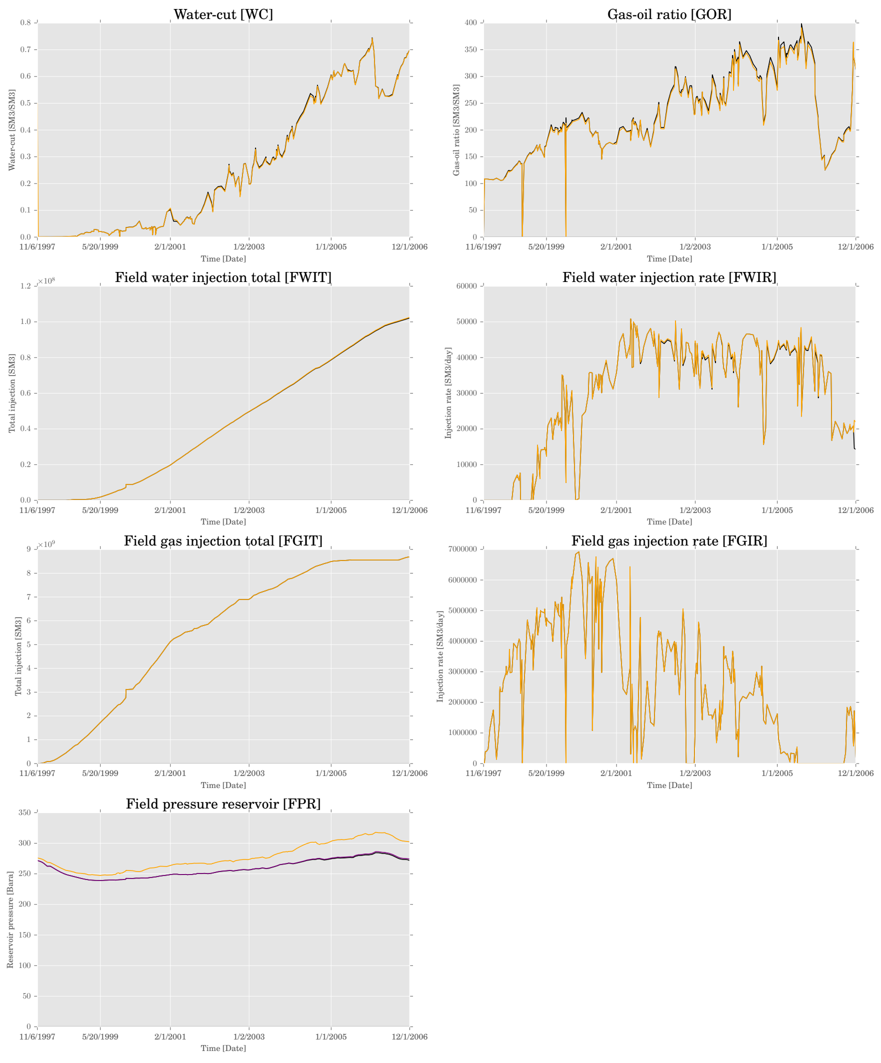


Figure D.3: Graphs for Eclipse [Black] and Flow [Orange] for Water-cut, Gas-oil ratio, water injection total, water injection rate, gas injection total, field gas injection rate and reservoir pressure ((Purple showing PAV from Flow .PRT log).

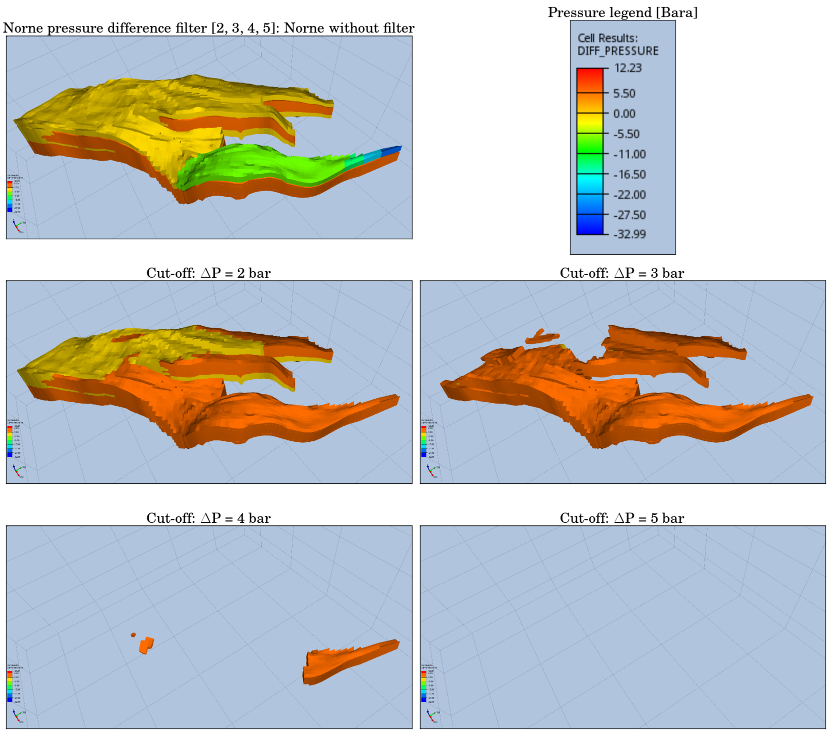


Figure D.4: Cell-by-cell pressure comparison cut-off at ΔP [2, 3, 4, 5].

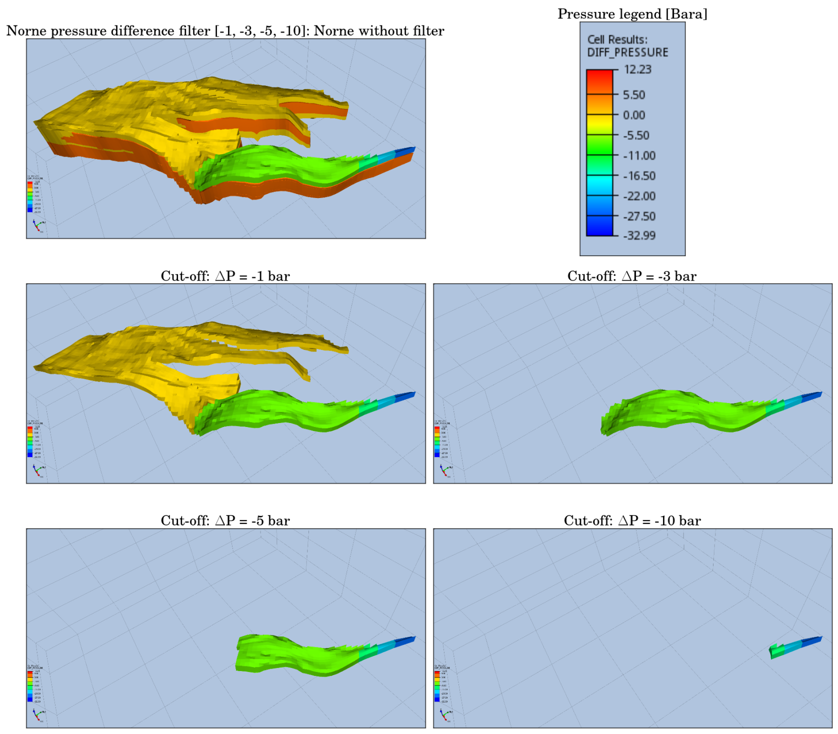


Figure D.5: Cell-by-cell pressure comparison cut-off at ΔP [-1, -3, -5, -10].

Appendix E

Saturation and Static Properties

This appendix contains the cell-by-cell difference in saturation profiles and static properties.

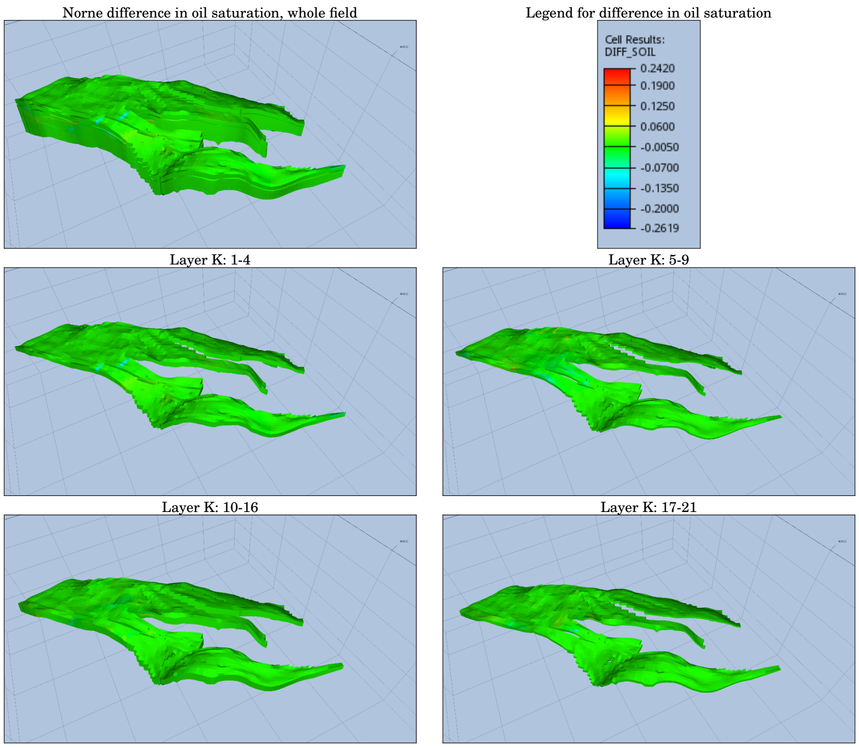


Figure E.1: Difference in oil saturation [ΔSOIL] for the last time step, through the layers in Norne. For the last report step. Positive values indicate Flow producing higher value, and negative values indicate that Flow produces reduced values compared with Eclipse.

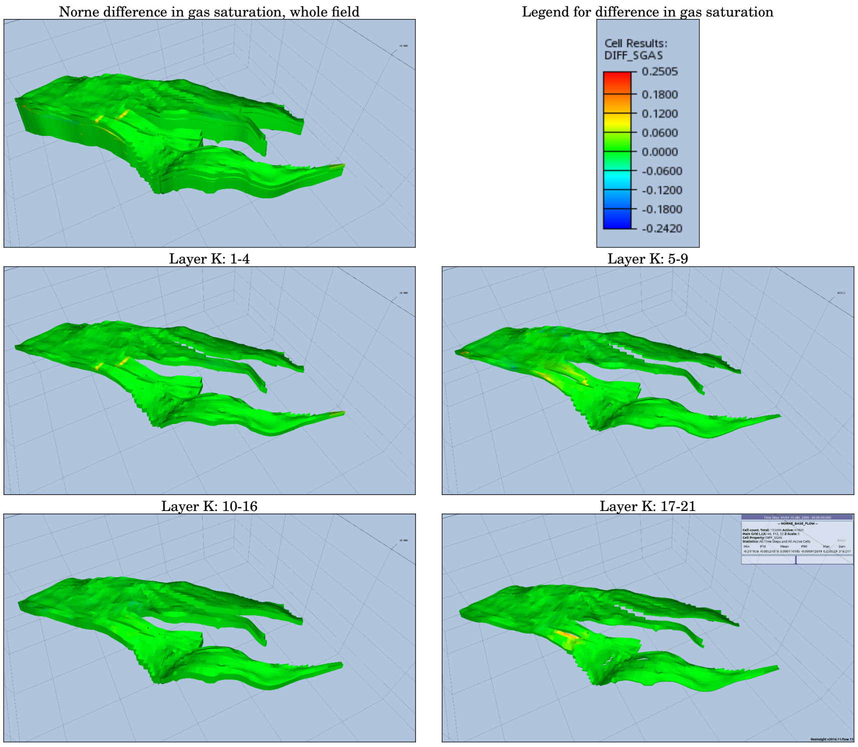
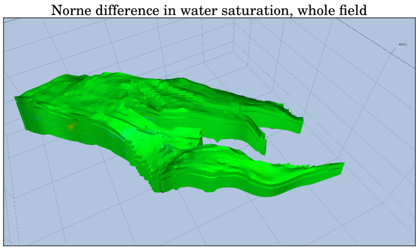
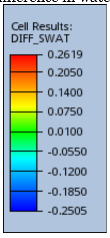


Figure E.2: Difference in gas saturation [ΔSGAS] for the last time step, through the layers in Norne. For the last report step. Red circle indicates area of interest. Positive values indicate Flow producing higher value, and negative values indicate that Flow produces reduced values compared with Eclipse.

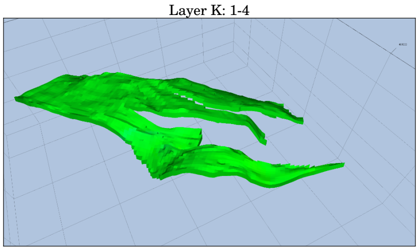
Norne difference in water saturation, whole field



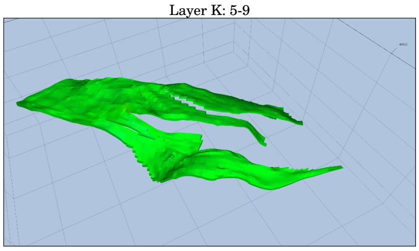
Legend for difference in water saturation



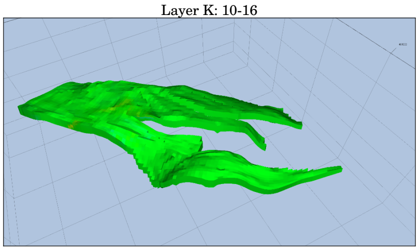
Layer K: 1-4



Layer K: 5-9



Layer K: 10-16



Layer K: 17-21

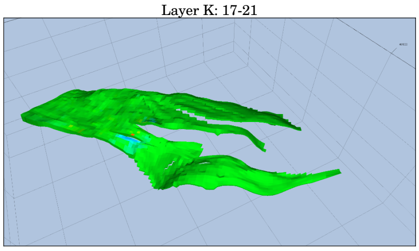


Figure E.3: Difference in water saturation [ΔSWAT] for the last time step, through the layers in Norne. For the last report step. Red circle indicates area of interest. Positive values indicate Flow producing higher value, and negative values indicate that Flow produces reduced values compared with Eclipse.

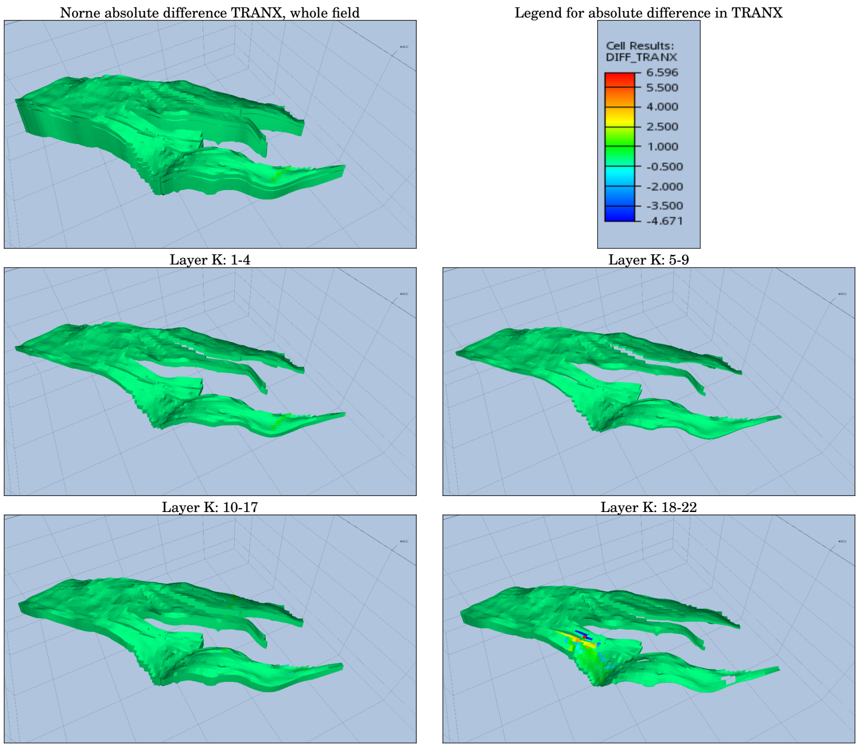


Figure E.4: Difference in transmissibility in the X-direction [ΔTRANX], Red circle indicates area of interest. Positive values indicate Flow producing higher value, and negative values indicate that Flow produces reduced values compared with Eclipse.

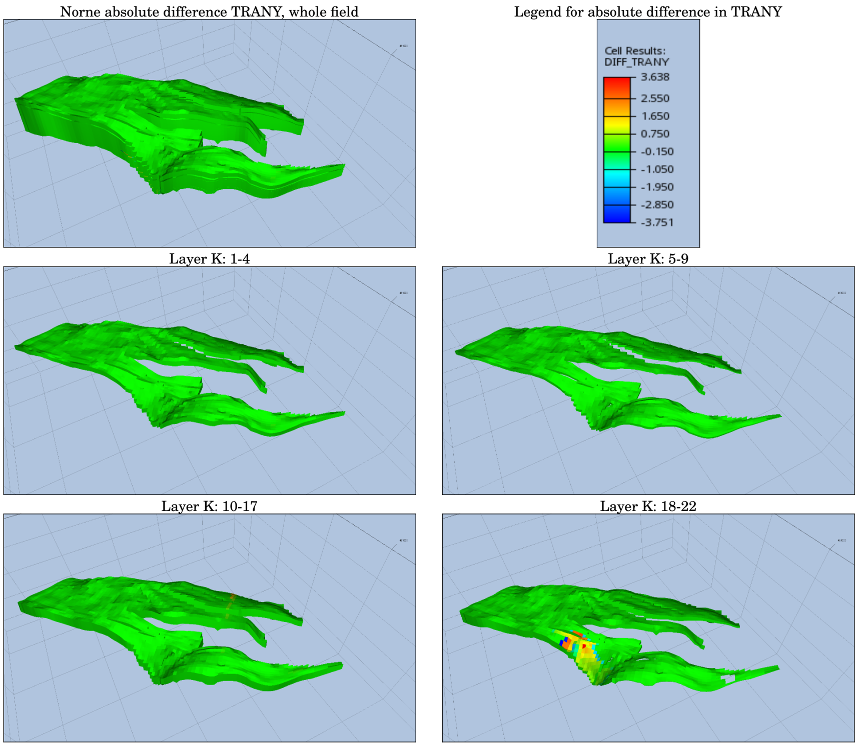


Figure E.5: Difference in transmissibility in the Y-direction [ΔTRANY], Red circle indicates area of interest. Positive values indicate Flow producing higher value, and negative values indicate that Flow produces reduced values compared with Eclipse.

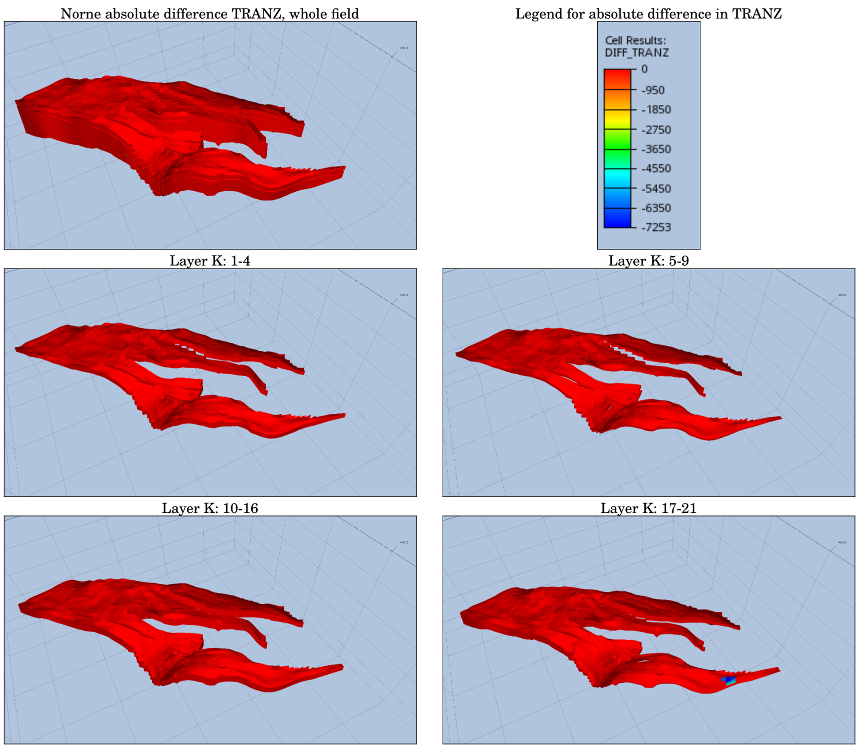


Figure E.6: Difference in transmissibility in the Z-direction [ΔTRANZ], Red circle indicates area of interest. Positive values indicate Flow producing higher value, and negative values indicate that Flow produces reduced values compared with Eclipse.

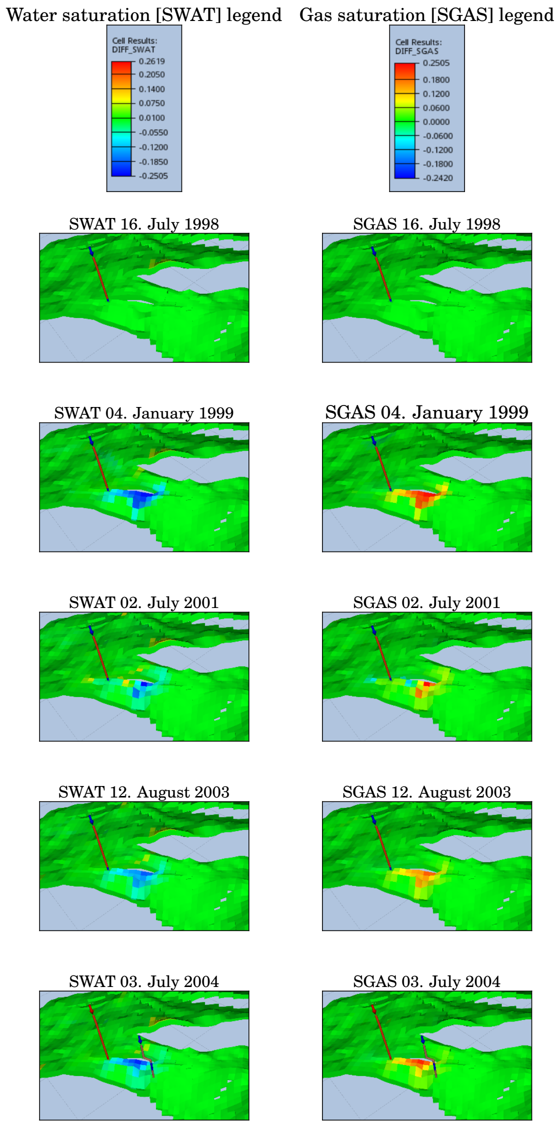


Figure E.7: Water and gas saturation through selected report steps. WAG injector C-1H is located at the left and WAG injector C-4AH is located on the right. Legends for water and gas saturation, at the top represents the difference from reference case [Flow−Eclipse].

Appendix F

References

Al-Kinani, Andreas, et al. "Adaptive Advisory Systems for Oil and Gas Operations." Integrated Operations in the Oil and Gas Industry: Sustainability and Capability Development, ed. Tom Rosendahl and Vidar Hepsø (2013): 262-284.

Aramco simulator name GIGAPOWERS: `http://www.oilandgasnewsworldwide.com/Article/33779/GigaPowers_delves_deep_into_reservoirs` (Retrieved 17.10.2016)

Bestpricecomputers.co.uk "Application Development (AppDev) Defined and Explained".. 2007-08-13. (Retrieved 30.11.16)

Biondi, Biondo. 3D seismic imaging. No. 14. Tulsa, Okla, USA: Society of Exploration Geophysicists, 2006.

Black Duck Software
`https://www.blackducksoftware.com/2016-future-of-open-source` (Retrieved 30.11.16)

Buchanan, D., & Huzcynski, A. (2010). Organizational behaviour: An introductory text (7th ed.). New York: Prentice Hall.

Burns, B. (2009). Managing change. Harlow, UK: Financial Times/Prentice Hall.

Carlile, Paul R. "Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries." Organization science15.5

(2004): 555-568.

Cummings, T., & Worley, C. (2009). Organization development & change. Mason, OH: South-Western.

Dalland, A., Worsley, D., & Ofstad, K. (Eds.). (1988). A lithostratigraphic scheme for the Mesozoic and Cenozoic and succession offshore mid-and northern Norway. Oljedirektoratet.

Distributed Unified Numerics Environment (DUNE): `https://dune-project.org`. Rretrieved 30-05-2017)

Edwards, Antony Roland, Oyvind Mydland, and Adolfo Henriquez. "The art of intelligent energy (iE)-Insights and lessons learned from the application of iE." SPE Intelligent Energy Conference and Exhibition. Society of Petroleum Engineers, 2010.

Eirik Morell. History matching of the Norne field. Masters thesis, Norwegian University of Science and Technology, September 2010.

Ensemble based Reservoir Tool `http://ert.nr.no/ert/index.php/Main_Page` (Retrieved 30.05.17)

Grimstad, Bjarne, et al. "On why model-based production optimization is difficult in the upstream industry." Published as a report in the IO center (2014).

Guldemond, Ewoud. Collaborative work environments in smart oil fields. PhD thesis, Radboud University Nijmegen, The Netherlands, 2011.

Hajizadeh, Yasin. "Ants can do history matching." SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers, 2010.

Henderson, John, Vidar Hepsø, and Øyvind Mydland. "What is a capability platform approach to integrated operations." An introduction to key concepts. Integrated Operations in the Oil and Gas Industry: Sustainability and Capability Development (2013).

Ivar Steffensen and Per Ivar Karstad. Norne Field Development: Fast Track From Discovery to Production. The Society of Petroleum Engineers, Statoil, April 1996.

Jansen, J. D., Brouwer, R., & Douma, S. G. (2009, January). Closed loop reservoir management. In SPE Reservoir Simulation Symposium. Society of Petroleum Engineers.

Kilamo, Terhi, et al. "From proprietary to open sourceGrowing an open source ecosystem." Journal of Systems and Software 85.7 (2012): 1467-1478.

Kristoffersen, Brage S. "Integrated Operations in Reservoir Engineering and Management: Software, trust, and development" (2016)

Landmark, Haliburton `https://www.landmark.solutions/Nexus-Reservoir-Simulation`
(Retrieved 11.12.2016)

Landrø, Martin. "Repeatability issues of 3-D VSP data." Geophysics 64.6 (1999): 1673-1679.

Landrø, Martin 2005 Seismic Reservoir Monitoring, Lecture notes, Norwegian University of science and technology, NTNU, Norway.

Larsen, Sjur. "Managing team leadership challenges in integrated operations." Integrated Operations in the Oil and Gas Industry. Sustainability and Capability Development. Hershey, PA: IGI (2012).

Larry Troan (2005). "Open Source from a Proprietary Perspective" (PDF). RedHat Summit 2006 Nashville. redhat.com. p. 10. Archived from the original (pdf) on 2014-01-22. Retrieved 2015-12-29.

Leonard-Barton, D. 1995. Well Springs of Knowledge: Building and Sustaining the Sources of Innovation. Harvard Business School Press, Boston, MA.

Mechell, Bryan James. "Understanding Patent Non-Assertion Agreements: The Enforceability of Microsoft's Open Specification Promise." AIPLA QJ 36 (2008): 179.

Minkoff, Susan E., et al. "Coupled geomechanics and flow simulation for time-lapse seismic modeling." Geophysics 69.1 (2004): 200-211.

Mockus, Audris, Roy T. Fielding, and James D. Herbsleb. "Two case studies of open source software development: Apache and Mozilla." ACM Transactions on Software Engineering and Methodology (TOSEM) 11.3 (2002): 309-346.

Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K., Ye, Y.,2002. Evolution pattern of open-source software systems and communities. In: IWPSE 02: Proceedings of the International Workshop on Principles of Software Evolution 2002. ACM. Press, pp. 7685.

Narduzzo, Alessandro, and Alessandro Rossi. Modularity in action: GNU/Linux and free/open source software development model unleashed. No. 020. Department of Computer and Management Sciences, University of Trento, Italy, 2008.

Oljenringens Landsforbund (OLF) / Norwegian Oil industry Association "Integrated Work Processes: Future work processes on the Norwegian Continental

Shelf" 20.10.2005.

OpenEarth Community (OEC), http://www.openearth.community (Retrieved 20.05.17)

Open Porous Media (OPM) Initiative, http://opm-project.org/ (Retrieved 30.05.17)

Open Source Initiative (OSI), https://opensource.org/osd-annotated (Retrieved: 30.11.16)

Raymond, Eric. "The cathedral and the bazaar." Knowledge, Technology & Policy 12.3 (1999): 23-49.

Rosendahl, Tom, et al. "Integrated Operations from a Change Management Perspective." (2013).

Rwechungura, Richard Wilfred, et al. "The Norne Field Case-A Unique Comparative Case Study." SPE Intelligent Energy Conference and Exhibition. Society of Petroleum Engineers, 2010.

Rwechungura, Richard Wilfred, Mohsen Dadashpour, and Jon Kleppe. "Advanced history matching techniques reviewed." SPE Middle East Oil and Gas Show and Conference. Society of Petroleum Engineers, 2011.

Saltelli, Andrea, et al. Sensitivity analysis in practice: a guide to assessing scientific models. John Wiley & Sons, 2004.

Schlumberger (SLB), The Future of Reservoir Management https://www.slb.com/~/media/Files/resources/mearr/num5/reservoir_mangement.pdf (Last checked 30.11.16)

Schryen, Guido. "Is open source security a myth?." Communications of the ACM 54.5 (2011): 130-140.

Technology Conversation
https://technologyconversations.com/2013/12/11/black-box-vs-white-box-testing/ (Retrieved 10.10.2016)

Tip Top Security: https://tiptopsecurity.com/safest-web-browser-chrome-firefox-ie-opera-safari-comparison-chart/ (Retrieved 30.11.16)

West, Joel, and Scott Gallagher. "Challenges of open innovation: the paradox of firm investment in open-source software." R&d Management 36.3 (2006): 319-331.

Wikipedia: Comparison of open-source and closed-source software `https://en.wikipedia.org/wiki/Comparison_of_open-source_and_closed-source_software` (Retrieved 08.12.2016)

Wikipedia: Simulation `https://en.wikipedia.org/wiki/Simulation` (Retrieved 11.12.2016)

Wikipedia: Software development `https://en.wikipedia.org/wiki/Software_development` (Retrieved 11.12.2016)

Zhu, Kevin Xiaoguo, and Zach Zhizhong Zhou. "Research note-lock-in strategy in software competition: Open-source software vs. proprietary software." Information Systems Research 23.2 (2012): 536-545.