Contents

[1. Dokumentasjon av analysearbeid 2](#_Toc119068360)

[1.1. Cognite Data Fusion (CDF) 2](#_Toc119068361)

[1.2. Digital Grunnmur 2](#_Toc119068362)

[1.3. CDF assets 2](#_Toc119068363)

[1.4. CDF events 3](#_Toc119068364)

[1.5. CDF hierarki 4](#_Toc119068365)

[1.6. A&E fra kontrollsystem i CDF (ABB – Ula/Valhall) 5](#_Toc119068366)

[1.7. Alarmer (ABB – Ula/Valhall) 6](#_Toc119068367)

[ Potensiell videre filtrering av alarmer 9](#_Toc119068368)

[1.8. Kjøreplaner og -status for hovedutstyr 9](#_Toc119068369)

[1.9. Functional Location format på tagnavn 10](#_Toc119068370)

[ FLOC prefix 10](#_Toc119068371)

[ Subtag suffix 11](#_Toc119068372)

[2. Future Work 12](#_Toc119068373)

[2.1. Revidere oversikter over hovedutstyr 12](#_Toc119068374)

[2.2. Kvitterte, men fortsatt aktive alarmer 12](#_Toc119068375)

[ Utfordring 12](#_Toc119068376)

[ Løsning pr. 10.11.2022 13](#_Toc119068377)

[2.3. ‘Shelved’ alarmer 13](#_Toc119068378)

[2.4. Hendelser fra K-IMS (Kongsberg) og Siemens 13](#_Toc119068379)

[2.5. Start/Stopp hovedutstyr presentert i CrewLog 13](#_Toc119068380)

[3. BlockLog presentasjon i CrewLog 14](#_Toc119068381)

[3.1. Ivar Aasen 14](#_Toc119068382)

[3.2. Ula 14](#_Toc119068383)

[3.3. Valhall 14](#_Toc119068384)

[3.4. Skarv 14](#_Toc119068385)

[3.5. Alvheim 14](#_Toc119068386)

[3.6. Edvard Grieg 14](#_Toc119068387)

[4. Veien videre for å bringe verdi fra CDF til CrewLog (Second opinion) 15](#_Toc119068388)

[5. Vedlegg 17](#_Toc119068389)

[5.1. Python funksjoner for å finne FLOC fra events kildetag 17](#_Toc119068390)

[6. Stakeholders 19](#_Toc119068391)

CrewLog (shift handover)

Automatiske hendelser fra kontrollsystem

# Dokumentasjon av analysearbeid

## Cognite Data Fusion (CDF)

CDF er Cognites skytjeneste / datalake for lagring, kontekstualisering og behandling av all type data. Aker BP har ambisjoner om at all prosessdata på sikt skal være tilgjengelig i CDF, slik at en anvende CDF til analysearbeid, utvikle applikasjoner på tvers av assets eller på andre måter hente skape verdi.

## Digital Grunnmur

Det pågår et arbeid med å frigjøre data fra kontrollsystemene offshore og få disse opp i CDF, i et prosjekt med navn Digital Grunnmur. Pr. 08.11.2022 er det fortsatt bare Ula og Valhall (ABB) som har fått på plass sin digitale grunnmur og har rådata fra kontrollsystemene tilgjengelig i CDF, men rundt årsskifte vil også Skarv og Alvheim (Kongsberg) være på plass, Ivar Aasen (Siemens) er planlagt før mars 2023.

## CDF assets

Et asset i CDF må ikke forveksles med det vi kaller et asset i Aker BP. Et asset i Aker BP er et oljefelt/område hvor vi har en eller flere oljeinstallasjoner som produserer olje og gass, f.eks. Valhall-assetet som består av Valhall feltsenter og flere flanke-plattformer, og Skarv-assetet som kun består av en FPSO.

Et asset i CDF derimot, er typisk et objekt som holder på informasjon som beskriver objektet i en eller annen grad. Det kan f.eks. være en ventil ute i anlegget, med informasjon som beskriver ventilen og tilhørende tidsserier som f.eks. viser ventilåpningen over tid, eller ei pumpe, motor eller lignende. Assets i CDF kan også være underobjekter til et annet objekt, f.eks. motoren til ei pumpe eller et statusobjekt som beskriver bestemte verdier som tilhører objektet, f.eks. disse 4 assetene:

* Lubeoil pumpe (23-PC-9111)
  + Motor (23-PC-9111-M01)
    - PSD trip shutdown signal (23-PC-9111-M01-24)
    - Running/stopped signal (23-PC-9111-M01-39)

Alle CDF assets består alltid av følgende informasjon/attributter som kan være av interesse:

|  |  |
| --- | --- |
| **INFO/ATTRIBUTT** | **BESKRIVELSE** |
| external\_id | Ekstern ID som gjerne kommer fra kilden bak assetet |
| name | Navnet på assetet (typisk tagnavn) |
| parent\_id | CDF ID til foreldre-assetet i CDF (rot-assets har ikke dette) |
| parent\_external\_id | Ekstern ID til foreldre-asset (rot-assets har ikke dette) |
| description | Beskrivelse av assetet |
| data\_set\_id | ID-en til CDF datasettet som assetet ligger i |
| metadata | Inneholder et sett med attributter unikt for assetet |
| source | Kilden bak opprettelsen av assetet (f.eks. Aveva eller OPC UA) |
| labels | Tittel som ofte beskriver kilden |
| id | Unik CDF ID for assetet |
| created\_time | Tidspunkt for når assetet ble opprettet i CDF |
| last\_updated\_time | Tidspunkt for når assetet sist ble oppdatert i CDF |
| root\_id | CDF ID til rot-assetet |

Merk spesielt at assets i CDF har en hierarkisk relasjon til hverandre og at hvert asset inneholder informasjon om hvilket asset de tilhører: **asset.parent\_id**

## CDF events

Eventer i CDF er en felles benevnelse for ulike hendelser med en kobling til et asset. Eventer kan f.eks. være at det har blitt opprettet en arbeidsordre for vedlikehold på et asset (f.eks. på ei pumpe), isoleringer, eller at det har blitt registrert en alarm eller event i kontrollsystemet på assetet.

Alle CDF events består alltid av følgende informasjon/attributter som kan være av interesse:

|  |  |
| --- | --- |
| **INFO/ATTRIBUTT** | **BESKRIVELSE** |
| external\_id | Ekstern ID som gjerne kommer fra kilden |
| data\_set\_id | ID-en til CDF datasettet som eventet ligger i |
| start\_time | Starttidspunkt for eventet |
| end\_time | Sluttidspunkt for eventet |
| type | Typebetegnelse for eventet |
| description | Beskrivelse av eventet (ikke alltid fullt beskrivende) |
| metadata | Inneholder et sett med attributter unikt for eventet, se kap. «A&E fra kontrollsystem i CDF» |
| asset\_ids | Liste over CDF ID-ene til assetene som eventet har en kobling til |
| id | Unik CDF ID for eventet |
| last\_updated\_time | Tidspunkt for når eventet sist ble oppdatert i CDF |
| created\_time | Tidspunkt for når eventet ble opprettet i CDF |

Merk spesielt at eventer i CDF har en relasjon til hvilke assets eventene stammer fra: **event.asset\_ids**

I dette prosjektet er alarmer og eventer (A&E) fra kontrollsystemene av interesse. Vi må passe på å skille mellom eventer fra kontrollsystem og eventer i CDF. Både alarmer og eventer fra kontrollsystemet blir omtalt som events når de kommer inn i CDF, men de inneholder typisk på metadata som man kan tyde for å skille mellom om det faktisk er en alarm eller et event.

## CDF hierarki

Assets i CDF har en relasjon til hverandre og relasjonene er oppbygde som et hierarki. Eventer har også en relasjon til et eller flere assets.

Eksempel på typisk hierarkioppbygning for dette prosjektet:

* **CDF ROOT ASSET**

name: VFN

data\_set\_id: 6810015115223364

source: aveva

id: 1202379980374596

…

* + …
    - **CDF ASSET**

name: 47-PS-12001-M01-01

parent\_id: 6877709216836998

data\_set\_id: 6810015115223364

source: aveva

id: 482220262989013

root\_id: 1202379980374596

…

* + - * **CDF ASSET**

name: 47-PS-12001-M01-01

parent\_id: 482220262989013

data\_set\_id: 140572846698809

source: OPC\_UA

id: 812863580834318

root\_id: 1202379980374596

…

* + - * + **CDF EVENT**

data\_set\_id: 140572846698809

asset\_ids: [812863580834318]

id: 2592908100484675

…

Som en ser av hierarkiet ovenfor så er det mulig å traversere oppover i asset-hierarkiet fra et CDF event. Asset ID-en som ligger i eventets **asset\_ids** attributt peker til ID-en til assetet over. Videre ser vi at assetets **parent\_id** attributt peker videre til ID-en til assetet over der igjen. Alle assets har også et **root\_id** attributt som peker til rot-assetets ID.

Merk også at vi her har to assets med samme navn (47-PS-12001-M01-01), men med ulike kilder – ett som stammer fra aveva og ett som stammer fra OPC\_UA. Dette skyldes at det meste av hierarkiet i CDF i utgangspunktet ble bygd opp av hierarkiet i Aveva, og når digital grunnmur ble implementert så inneholdte datastrømmen fra kontrollsystemene (OPC UA) objekter med samme navn. For å ivareta informasjonen fra begge kilder (Aveva og OPC UA) så har de fått hvert sitt unike asset, men OPC UA assetet har fått en relasjon til Aveva assetet for å ivareta hierarkistrukturen. Eventet, som også kommer fra OPC UA, har selvsagt fått relasjon til OPC UA assetet. Merk også at OPC UA eventet og OPC UA assetet tilhører samme datasett, mens de to Aveva assetene tilhører et annet datasett.

## A&E fra kontrollsystem i CDF (ABB – Ula/Valhall)

A&E fra kontrollsystemene kommer til CDF fra den digitale grunnmuren via OPC UA. All OPC UA data ligger i hver sine lokasjonsspesifikke datasett i CDF:

|  |  |
| --- | --- |
| **CDF datasett** | **data\_set\_id** |
| OPC UA data VAL | 140572846698809 |
| OPC UA data ULA | 2086908079872503 |

Som nevnt tidligere så lagres både alarmer og eventer (A&E) fra kontrollsystemene som events i CDF, og vi må derfor kunne skille mellom hva som er alarmer og hva som er events. Det er i hovedsak i metadataene til et CDF event vi finner beskrivende informasjon om eventet og kan tyde slike ting.

Alle eventer i CDF som kommer fra A&E består alltid av følgende metadata/attributter som kan være av interesse:

|  |  |
| --- | --- |
| **MATADATA/ATTRIBUTT** | **BESKRIVELSE** |
| eventCategory | A&E kategori nummer fra kontrollsystem |
| eventCategoryName | A&E kategori navn fra kontrollsystem, samsvarer med *eventCategory* |
| eventCounter | A&E løpenummer fra kontrollsystem |
| EventData | Duplikat av all metadata fra kontrollsystem |
| eventType | A&E type fra kontrollsystem |
| id | A&E id fra kontrollsystem |
| message | A&E beskrivelse (*event.description* hentes herfra) |
| severity | A&E alvorlighetsgrad |
| Severity | A&E alvorlighetsgrad (identisk med severity) |
| source | A&E kilde (systemtag fra kontrollsystem) |
| SourceName | A&E kilde (virker å være identisk med source) |
| time | A&E tidsstempel (for når det spesifikke A&E skjedde) |

Vi kan bruke **severity**-attributtet (*event.metadata.severity*) til å skille mellom alarmer og eventer, ref. alarm FDS-ene *G00-23-IN-5067-00* (Ula) og *VAL-AC-I-0212* (Valhall):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SYSTEM** | **PRIORITY 1** | **PRIORITY 2** | **PRIORITY 3** | **PRIORITY 4** | **EVENT** |
| **ESD** | 901 | 801 | 701 | 601 | 401 |
| **F&G** | 902 | 802 | 702 | 602 | 402 |
| **PSD** | 903 | 803 | 703 | 603 | 403 |
| **PCS (kun valhall)** | 904 | 804 | 704 | 604 | 403 |
| **HVAC (kun valhall)** | 905 | 805 | 705 | 605 | 405 |
| **SCMS** | 906 | 806 | 706 | 606 | 406 |
| **AC400** | 909 | 809 | 709 | 609 | 409 |

CDF events med en severity på 401-406 eller 409 er altså eventer fra kontrollsystemet, mens alarmer har severity som begynner på 6, 7, 8, eller 9, avhengig av hvilken prioritet alarmen har.

I tillegg er det identifisert følgende metadata/attributter som A&E tidvis består av, som også kan være av interesse (noen finnes kun for alarmer og ikke eventer):

|  |  |  |  |
| --- | --- | --- | --- |
| **METADATA/ATTRIBUTT** | **BESKRIVELSE** | **A** | **E** |
| ackRequired | Om alarmen krever å bli kvittert eller ikke | X |  |
| activeTime | Tidspunktet for når alarmen først ble aktiv | X |  |
| actorId | Bruker som gjorde en handling ifm. alarmen | X |  |
| alarmState | Samme som *AlarmStatus*, bare sjeldent brukt | X |  |
| AlarmState | Alarmstatus (aktiv, returned-to-normal/inaktiv, etc.) | X |  |
| class | Samme som *Class*, bare sjeldent brukt | X | X |
| Class | Beskriver systemnode og system, ref. dok. VAL-AC-I-0231 | X | X |
| conditionName | Beskrivende alarmtekst (bør komme i tillegg til *message*) | X |  |
| subConditionName | Virker å være identisk med *conditionName* | X |  |
| newState | Tilstand for om alarm er aktiv og/eller kvittert | X |  |
| newStateText | Beskrivende tekst for *newState* | X |  |
| objectDescription | Samme som *ObjectDescription*, bare sjeldent brukt | X | X |
| ObjectDescription | Beskrivelse på objektet som A&E stammer fra | X | X |
| processSection | Samme som *ProcessSection*, bare sjeldent brukt | X | X |
| ProcessSection | Beskriver prosessområde, ref. dok. VAL-AC-I-0231 | X | X |

## Alarmer (ABB – Ula/Valhall)

Som nevnt så kan vi skille mellom hva som er alarmer og hva som er eventer fra kontrollsystemet med **severity**-attributtet, men i tillegg til dette kan severity brukes til å skille mellom hvilken prioritet den gitte alarmen har.

|  |  |  |  |
| --- | --- | --- | --- |
| **PRIORITY 1** | **PRIORITY 2** | **PRIORITY 3** | **PRIORITY 4** |
| Sikkerhetskritiske alarmer | Eskalerende alarmer | Ikke-eskalerende alarmer | Ikke-kritiske alarmer |

Det mest åpenbare er kanskje å filtrere alarmer på prioritet 1 og 2, som begge krever rask handling fra operatørene, og er alarmer som kan være relativt viktige å kjenne til.

Alarmer kan filtreres videre på **newState**-attributtet, hvor:

|  |  |  |
| --- | --- | --- |
| **newState** | **newStateText** | **FORKLARING** |
| **1** | Enabled | Alarmårsaken har returnert til normaltilstand, men alarmen er fortsatt ikke blitt kvittert |
| **3** | Enabled, Active | Ny aktiv alarm |
| **5** | Enabled, Acknowlegded | Alarmen er blitt kvittert og årsaken har returnert til normaltilstand |
| **7** | Enabled, Active, Acknowledged | Alarmen er blitt kvittert, men årsaken er fortsatt aktiv |

Alle nye alarmer skal altså først komme med en newState på 3, og etter hvert gå til 5, som betyr at alarmen er blitt kvittert og alarmårsaken har gått tilbake til normaltilstand. Alarmer skal altså bevege seg fra 3 til 5 og kan enten bevege seg 3 – 7 – 5 eller 3 – 1 – 5, alt etter som hva som skjer først av alarmkvittering og returnering til normaltilstand.

Rent praktisk betyr dette at én og samme alarm kan dukke opp i CDF som flere CDF events, først ett event med newState 3, deretter et nytt event med f.eks. newState 7, og så et siste event med newState 5.

Alarmer kan også filtreres på **alarmState/AlarmState**-attributtet, hvor:

|  |  |  |
| --- | --- | --- |
| **alarmState**  **/AlarmState** | **BESKRIVELSE** | **FORKLARING** |
| **ACT** | Active | Alarmtilstanden er aktiv (og har enten *newState* 3 eller 7) |
| **RTN** | Return-to-normal | Alarmtilstanden har returnert til normalen og er ikke lengre aktiv (har enten *newState* 1 eller 5) |
| **ABL** | Automatically blocked | Alarmen har blitt automatisk blokkert av systemet (kan ha hvilken som helst *newState*) |
| **SLV** | Shelved | Alarmen er blitt lagt på hyllen (kan ha hvilken som helst *newState*) |

Etter en mapping av alarmer på **newState (n)** og **alarmState/AlarmState (a)** så finner vi følgende sammenhenger (utdrag av alarmer fra hovedutstyr):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ULA** | | | | |  | **VALHALL** | | | |
| **n\a** | **ACT** | **RTN** | **ABL** | **SLV** |  | **ACT** | **RTN** | **ABL** | **INGEN** |
| **1** | 0 | 1325 | 208 | 9 |  | 0 | 11835 | 319 | 5300 |
| **3** | 2613 | 0 | 54 | 15 |  | 11725 | 0 | 34 | 1323 |
| **5** | 0 | 2156 | 34 | 4 |  | 0 | 4798 | 18 | 648 |
| **7** | 1904 | 0 | 33 | 3 |  | 3570 | 0 | 28 | 384 |

Som nevnt så kommer nye alarmer først med newState 3. Hvis vi velger å filtrere på eventer med newState 3 så skal i teorien resterende eventer med andre newState-tilstander være de samme alarmene, bare med en oppdatert tilstand. Ved å filtrere på flere enn én newState-tilstand risikerer vi altså å få samme alarmtilfelle flere ganger.

Når det gjelder alarmState/AlarmState så er alarmer som blir blokkert automatisk av systemet (ABL) ikke viktige for en operatør og derfor ikke av interesse. Shelved alarmer (SLV) kan være av interesse, for å informere neste skift om hvilke alarmer som er «gjemt» på denne måten – mer om dette i kapittelet med fremtidige muligheter / future work.

Men om vi skal fokusere på alarmer som potensielt kan være identifikasjon på viktige hendelser ila. et skift, så foreslår jeg å filtrere på alarmer med newState 3 og alarmState/AlarmState ACT (eller ingen alarmstate/AlarmState, som tydeligvis mange alarmer på Valhall ikke har). Shelved alarmer har blitt lagt på hyllen av en grunn, og vil i seg selv ikke informere noen viktig hendelse. Det samme gjelder automatisk blokkerte alarmer.

Jeg foreslår derfor følgende filtrering for å identifisere potensielt interessante og/eller viktige alarmer:

* Severity 801-806, 809, 901-906, 909 (prioritets 1 og 2 alarmer)
* newState 3 (nye aktive alarmer)
* alarmState/AlarmState ACT eller ingen

Tabellen nedenfor viser et utdrag av de alarmene vi finner på Ula og Valhall over en periode på 60 skift (30 døgn) med filtrene ovenfor. For Ula er filteret newState 3 og alarmState/AlarmState ACT. Filteret for Valhall er newState 3 og alarmState/AlarmState ACT eller ingen. Som vi ser finner vi fortsatt mange alarmer med disse filtrene, og vi har et snitt på 108,8 og 133,8 alarmer pr. skift på henholdsvis Ula og Valhall. For en operatør som ønsker en oversikt over de viktigste hendelsene ila. skiftet sitt så vil et så høyt antall sannsynligvis bare virke overveldende og fremstå som støy. Overnevnte filter er med andre ord ikke tilstrekkelig og vi bør få redusert søket ytterligere.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **ULA** | | | **VALHALL** | | |
| **SYSTEM** | **PRIORITY 1** | **PRIORITY 2** | **TOT(AVG)** | **PRIORITY 1** | **PRIORITY 2** | **TOT(AVG)** |
| **ESD** | **7** (0,1) | **90** (1,5) | **97** (1,6) | **68** (1,1) | **26** (0,4) | **94** (1,6) |
| **F&G** | **237** (4,0) | **1838** (30,6) | **2075** (34,6) | **729** (12,2) | **543** (9,1) | **1272** (21,2) |
| **PSD** | **15** (0,3) | **55** (0,9) | **70** (1,2) | **1** (0,02) | **517** (8,6) | **518** (8,6) |
| **PCS** | **122** (2,0) | **440** (7,3) | **562** (9,4) | **24** (0,4) | **3558** (59,3) | **3582**(59,7) |
| **HVAC** | N/A | N/A | N/A | **0** (0,0) | **179** (3,0) | **179** (3,0) |
| **SCMS** | N/A | N/A | N/A | **6** (0,1) | **16** (0,3) | **22** (0,4) |
| **AC400** | **280** (4,7) | **3446** (57,4) | **3726** (62,1) | **599** (10,0) | **1763** (29,4) | **2362** (39,4) |
| **TOT(AVG)** | **661** (11,0) | **5869** (97,8) | **6530** (108,8) | **1427** (23,8) | **6602** (110,0) | **8029**(133,8) |

Et naturlig steg videre er gjerne å filtrere på hvor alarmene kommer fra, f.eks. overordnet filtrering på system (del av *event.severity*), eller mer direkte filtrering på alarmkilde (*event.source*). Min antakelse er at alle system gjerne inneholder kritisk utstyr hvor viktige hendelser kan forekomme, slik at vi ikke bør filtrere på system, men heller filtrere på alarmkilder.

Filtrering på alarmkilder er dessverre nokså mer krevende, da vi trenger en liste over hvilket utstyr vi ønsker alarmer fra, f.eks. en liste over hovedutstyr, stort/tungt utstyr, kritisk utstyr eller lignende. Jeg lagde i tidlig fase lister over slikt utstyr for samtlige Aker BP assets basert på oversikter de hadde, f.eks. kjøreplanoversikter. Disse er kun blitt brukt for «proof-of-concept» og er på ingen måte perfekte, og hvis dette konseptet viser seg verdifullt bør disse revideres for å luke ut irrelevant utstyr og evt. legge til manglende utstyr.

Ved å filtrere på nåværende liste over hovedutstyr får vi redusert antallet alarmer betraktelig. Merk at følgende tabell er nå over en periode på 659 skift og ikke 60 som den forrige. I tillegg er tallene i parentesene nå snittet pr. «*skifttilfelle*» med respektive alarmer. F.eks. så har det på Ula i løpet av 659 skift kommet totalt 142 prioritet 1 og 2 F&G alarmer (med newState 3 og alarmState ACT). Disse var fordelt på 100 skifttilfeller, slik at det i snitt kom 1,42 slike alarmer pr. skifttilfelle med den typen alarmer. Den respektive totalen på 553 og 367 på Ula og Valhall var fordelt på henholdsvis 183 og 168 skifttilfeller, slik at det i snitt kom 3,0 og 2,2 alarmer pr. skifttilfelle med det valgte filteret og hovedutstyret, i motsetning til 108,8 og 133,8 alarmer pr. *skift* uten filtrering på hovedutstyr. **Når ønsket er å redusere mengden alarmer til et overkommelig og tilstrekkelig nivå er det med andre ord ekstremt effektivt å filtrere på alarmkilde, men jobben ligger da i det å definere hvilket utstyr vi ønsker alarmer fra.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **ULA** | | | **VALHALL** | | |
| **SYSTEM** | **PRIORITY 1** | **PRIORITY 2** | **TOT(AVG)** | **PRIORITY 1** | **PRIORITY 2** | **TOT(AVG)** |
| **ESD** | **0** (0) | **0** (0) | **0** (0) | **3** (3,0) | **0** (0) | **3** (3,0) |
| **F&G** | **125** (1,3) | **17** (2,1) | **142** (1,4) | **2** (2,0) | **1** (1,0) | **3** (1,5) |
| **PSD** | **0** (0) | **0** (0) | **0** (0) | **0** (0) | **41** (3,2) | **41** (3,2) |
| **PCS** | **0** (0) | **177** (12,6) | **177** (12,6) | **57** (1,6) | **130** (1,6) | **187** (1,7) |
| **HVAC** | N/A | N/A | N/A | **12** (3,0) | **1** (1,0) | **13** (2,6) |
| **SCMS** | N/A | N/A | N/A | **0** (0) | **20** (6,7) | **20** (6,7) |
| **AC400** | **67** (2,4) | **167** (2,2) | **234** (2,4) | **45** (1,8) | **55** (1,6) | **100** (1,8) |
| **TOT(AVG)** | **192** (1,6) | **361** (3,9) | **553** (3,0) | **119** (1,8) | **248** (2,1) | **367** (2,2) |

Tabellen er etter min mening proof-of-concept for at filtrering på alarmkilde kan benyttes, og hvis vi bestemmer oss for å benytte oss av dette så bør vi revidere listene over hovedutstyr først.

### Potensiell videre filtrering av alarmer

Nedenfor har jeg nevnt noen måter jeg har identifisert som man potensielt kan filtrere alarmene videre på.

#### eventCategory/eventCategoryName

Attributtene *eventCategory* og *eventCategoryName* kan være med på å beskrive tilstanden til en alarm, ref. OPC A&E, f.eks.:

* *eventType*: ‘Condition’ (eventType er som regel Condition for alarmer)
  + *eventCategoryName*: ‘Process Condition Event’
    - *conditionName*: f.eks. ‘High High’, ‘Fault’, ‘Warning’, etc.
  + *eventCategoryName*: ‘Process(MB300)’
    - *conditionName*: f.eks. ‘LIM L1’, ‘Value’, ‘MV > H1’, etc.

Attributtet *conditionName* kan man mulig også filtrere på, men dette attributtet virker å holde på en hel rekke forskjellige verdier som ikke nødvendigvis fremstår standardisert.

#### CDF trestruktur

Cognites SDK har *asset\_subtree\_ids* som en alternativ parameter man kan bruke når man henter events. Jeg har ikke gravd i dette noe særlig, men jeg innbiller meg at man kan hente eventer fra en hel gren om man kjenner til ID-en til grenroten, og da med andre ord hente events fra alt utstyr tilhørende et hovedutstyr, men det gjenstår å finne ut av.

## Kjøreplaner og -status for hovedutstyr

Kjøreplaner er noe alle Aker BP assets benytter seg av for redundant hovedutstyr. Redundans er nødvendig for å f.eks. kunne opprettholde produksjon om et utstyr feiler eller trenger vedlikehold, og når man først har redundant utstyr er det naturlig å ha kjøreplaner for slikt utsyr for å bl.a. avlaste det og redusere slitasje.

Slike kjøreplaner har typisk info om stort redundant utstyr som parallelle pumper, motorer, generatorer, etc., og hvilket utstyr (f.eks. pumpe A eller B) som til enhver tid kjører, og hvilket som er satt ut av drift. Det kan for eksempel være at pumpe A går i oddetallsmåneder og pumpe B skal gå i partallsmåneder. Slike kjøreplaner er pr. dags dato veldig manuelle i Aker BP, og består gjerne bare av et Excel-ark som operatørene har rutiner for å holde oppdatert.

Med Digital Grunnmur vil det meste av rådata fra kontrollsystemene bli tilgjengeliggjort i CDF, slik at vi kan få tak i data om utstyr kjører eller ikke. Vi kan med andre ord få tak i kjørestatusen til stort redundant utstyr og forhåpentligvis anvende denne dataen på en verdifull måte inn mot CrewLog.

Hvis vi har kunnskap om nåværende kjørestatus for utstyret, og i tillegg informasjon om kjøreplanen til det respektive utstyret, så kan vi lage en veldig enkel logikk for å f.eks. varsle om feil:

|  |  |  |
| --- | --- | --- |
| **Kjører utstyret?** | **Skal utstyret kjøre?** | **Handling** |
| Ja | Ja | Ingen |
| Nei | Nei | Ingen |
| Nei | Ja | Opprett hendelse, f.eks. «Pumpe A følger ikke kjøreplan», e.l. |
| Ja | Nei | Opprett hendelse, f.eks. «Pumpe B skulle vært byttet med A», e.l. |

Alt en trenger informasjon om for denne enkle logikken er om utstyret kjører (fra CDF), og om utstyret er tiltenkt å kjøre (fra kjøreplan) for ethvert tidspunkt. Merk at det er viktig å legge inn et slingringsmonn i denne logikken, så en ikke ender opp med å generere hendelser når man er i ferd med eller holder på å gjennomføre utstyrsbytte. Dette kan lett unngås ved å implementere en tidsforsinkelse på f.eks. ett skift, eller 12 timer.

En idé kan også være å bygge videre på dette konseptet, og velge å opprette hendelser og/eller *oppgaver* som kan varsle om planlagt bytte av utstyr basert på kjøreplan:

|  |  |  |
| --- | --- | --- |
| **Kjører utstyret?** | **Skal utstyret kjøre *i morgen*?** | **Handling** |
| Ja | Ja | Ingen |
| Nei | Nei | Ingen |
| Nei | Ja | Opprett hendelse, f.eks. «Pumpe A vil byttes til B i morgen», e.l. |
| Ja | Nei | Opprett **oppgave**, f.eks. «Utfør bytte fra pumpe B til A», e.l. |

## Functional Location format på tagnavn

Alarmer og eventer (A&E) fra kontrollsystemet bør kunne linkes til en SAP Functional Location (FLOC), slik at en kan klikke seg inn på eller søke etter FLOC i SAP og få opp eventuelle notifikasjoner, arbeidsordrer o.l. på utstyret som alarmen/eventet kommer fra.

A&E fra kontrollsystemet blir lagret som eventer i CDF med ulike metadata-attributter, og blant disse har vi et attributt for kilden til den gitte alarmen/eventet (*source*). Formatet på kilden til A&E er dessverre ikke likt formatet til SAP Functional Location – en problemstilling vi bør finne en løsning på. Det mangler f.eks. prefix for maintenance plant(?)/plattform. I tillegg er kilden til mange alarmer og eventer fra «underobjekter» i kontrollsystemet eller «subtags» til hovedtaget.

F.eks. så kan kilden til et event være **Z-0501-01**, hvor **Z-0501** er hovedtaget, **01** er subtag for et underobjekt i kontrollsystemet som gir informasjon om hovedtagets start/stopp- eller åpen/lukket-tilstand. Subtags beskriver altså typisk kontrollsystemobjekter og det vil ikke nødvendigvis gi noen verdi til operatøren å søke opp disse i SAP, da notifikasjoner, arbeidsordrer o.l. gjerne er koblet mot det fysiske utstyret (hovedtaget), og ikke mot kontrollsystemobjekter.

Utfordringen blir altså å finne FLOC prefix, da dette ikke er tilgjengelig i metadataen til A&E, og i tillegg fjerne evt. unødvendige suffix (f.eks. subtag), slik at vi kan omdøpe A&E-kilden til riktig FLOC. Eksempelvis bør **Z-0501-01** omdøpes til **ULP-Z-0501**, hvor plattformkoden til Ula P, **ULP** (FLOC prefix), er lagt til og **01** er fjernet da dette er suffix for subtag/kontrollsystemobjekt.

### FLOC prefix

Som nevnt så finnes ikke informasjon om plattformkoden i metadataen til eventene, men et rekursivt søk oppover i asset-hierarkiet viser seg å finne assets som holder på slik informasjon – for mange assets finner vi dette i metadata-attributtet ‘*PLATFORM CODE*’.Den letteste metoden jeg har identifisert for å finne plattformkoden er altså enkelt og greit å manøvrere seg oppover i asset-hierarkiet og lete etter første asset som holder på ‘*PLATFORM CODE*’-attributtet. Nedenfor følger et eksempel på hvordan et slikt rekursivt søk kan gjøres i Python:

event = <event fra CDF>  
temp\_asset = event.asset\_ids[0]  
prefix = temp\_asset.metadata.get(‘PLATFORM CODE’, ‘’)  
**while not** prefix **and** temp\_asset.parent\_id:  
 temp\_asset = temp\_asset.parent()  
 prefix = temp\_asset.metadata.get(‘PLATFORM CODE’, ‘’)  
floc\_prefix = prefix

### Subtag suffix

Hvis suffix er på formatet **–00** til **–99** eller **\_00** til **\_99** kan det være så enkelt som å fjerne de tre siste tegnene. Men hva om suffix består av flere elementer, f.eks. <hovedtag>**–M01–39**, skal vi fjerne alt?

Min løsning på dette problemet er å ta utgangspunkt i navnekonvensjonen til tags på de ulike Aker BP assetene. Jeg har også tatt utgangspunkt i at hovedtaget er det vi er ute etter, slik at også M01 må fjernes fra eksempelet ovenfor. Denne løsningen er nokså mye mer avansert å implementere, men jeg har lagt ved et eksempel på hvordan jeg har satt sammen funksjoner for dette i Python som vedlegg i dette dokumentet – se «Python funksjoner for å finne FLOC fra events kildetag» i «Vedlegg»-kapittelet. Funksjonen *get\_floc(event)* returnerer <FLOC-prefix>-<hovedtag>, eller <FLOC-prefix>-<kildetag> hvis den ikke klarte å finne hovedtag.

Jeg har kun lagd funksjoner for Ula og Valhalls tagkonvensjon, men andre Aker BP assets kan enkelt legges inn i *extract\_tag\_names()*-funksjonen. Tagformatene jeg har tatt utgangspunkt i er:

Ula: A(AAA)-NNNN(N)(A)-(A)

Valhall: NN-AA(A)-NNNN(NN)(A)

hvor A er bokstaver, N er tall og parantesene er usikre.

Eks. Ula: LSHH-4204, SE-1512-A, PM-0601B

Eks. Valhall: 25-HX-9105A, 11-FIT-170031

Funksjonene i kodeeksempelet er kanskje unødvendig avanserte for CrewLog da jeg har valgt å rekursivt søke etter et asset i CDF med samme navn for å forsikre meg om at hovedtag-et faktisk finnes (*get\_true\_parent()*-funksjonen). For CrewLog sin del vil jeg tro SAP har andre måter å verifisere dette på, slik at det bare er *extract\_tag\_names()*-funksjonen vi egentlig trenger å forholde oss til.

# Future Work

## Revidere oversikter over hovedutstyr

Nåværende lister er basert på diverse oversikter fra de ulike Aker BP assetene, f.eks. kjøreplaner. Disse er blitt brukt for «proof-of-concept», men er på ingen måte perfekte og bør revideres før funksjonalitet som avhenger av disse blir tatt i bruk.

## Kvitterte, men fortsatt aktive alarmer

Alarmer som er kommet i løpet av det siste skiftet og blitt kvittert, men hvor tilstanden/årsaken fortsatt ikke har returnert til normalen (newState=7) er gjerne alarmer som fortsatt krever en form for aksjon/handling før tilstanden er løst. Dette er alarmer som potensielt kan ha stor verdi ved skiftbytte, ved at påtroppende skift raskt får oversikt over de alarmene som trenger aksjon.

|  |  |  |
| --- | --- | --- |
| **newState** | **newStateText** | **FORKLARING** |
| **1** | Enabled | Alarmårsaken har returnert til normaltilstand, men alarmen er fortsatt ikke blitt kvittert |
| **3** | Enabled, Active | Ny aktiv alarm |
| **5** | Enabled, Acknowlegded | Alarmen er blitt kvittert og årsaken har returnert til normaltilstand |
| **7** | Enabled, Active, Acknowledged | Alarmen er blitt kvittert, men årsaken er fortsatt aktiv |

### Utfordring

Nye alarmer fra kontrollsystemet identifiseres som et nytt CDF event med severity >600 og newState=3. Når f.eks. tilstanden går tilbake til normalen eller alarmen blir kvittert så logges dette som et nytt event med en annen newState-verdi. Dvs. at én og samme alarm vil i praksis generere flere events med ulike newState-verdier, f.eks.:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ny alarm** | à | **Alarm blir kvittert** | à | **Tilstand går tilbake til normal** |
| newState=3 | newState=7 | newState=5 |

1. En utfordring er hvordan registrerer vi overgangen fra newState=7 til newState=5, dvs. hvordan kan vi identifisere relasjonen mellom de to ulike eventene?

2. Hvordan vi skal registrere dette i loggen i CrewLog er også en utfordring. Ønsket er jo at loggen til enhver tid skal samsvare med virkeligheten, slik at en kan stole på dataen, og loggen må da til enhver tid være oppdatert. Hvordan skal vi forholde oss til alarmer som har newState=7 på oppdateringstidspunktet, men som returnerer til normaltilstand før skiftet er ferdig (neste oppdateringstidspunkt? Skal vi generere hendelser av alle alarmer med newState=7 underveis (for å ivareta samsvar med virkeligheten), og så oppdatere/slette respektive hendelser om de har gått til newState=5 på et senere tidspunkt?

### Løsning pr. 10.11.2022

Vi har satt opp et kall som samler inn alle CDF event med severity >600 og newState=3. Hvis newState=5 eller newState=1 eventer er aktivert for samme Tag som allerede er hentet i kallet med newState=3 så slettes disse. Kallet bruker også Hovedutstyret for ULA som referanse.

Resultatet er at vi har tilgjengelig alle alarmer som fortsatt er aktive til enhver tid «Stående alarmer»

Drawback: Jeg har sammenlignet med KPI alarm rapporten fra Ula med stående alarmer og det er kun 23 samsvarende alarmer av de 86 unike alarmene vi har fra CDF, dette legges på is.

## ‘Shelved’ alarmer

Alarmlisten for Shelved alarms bør være tilgjengelig i CrewLog.

Det er ulikt mellom installasjonene hvordan blir brukt og om tilgjengelig. Det er planer om standardisering av hvordan «Shelved» alarmer skal håndteres på tvers av alle installasjoner i Aker BP

## Hendelser fra ABB, K-IMS (Kongsberg) og Siemens

I og med at vi besluttet å ikke gå videre med full integrasjon med IMS-systemene vil vi ikke få dataflyt fra K-IMS til CrewLog, og derfor ingen mulighet til å generere hendelser i CrewLog basert på opprettede hendelser i K-IMS.

Full integrasjon med IMS-systemer (gjerne via CDF) har her et verdipotensial vi ikke får realisert på nåværende tidspunkt.

Cognite skal tilgjengeliggjøre data fra Skarv og Alvheim i CDF februar 2023

For Ivar Aasen skal Cognite være klar med data i CDF senest mars 2023.

## Start/Stopp hovedutstyr presentert i CrewLog

Arbeid pågår for å identifisere start/stopp på VAL og ULA

Møte med Ken Roger 18.11.22 avholdt, Ken Roger sender over Tagliste på de aktuelle TAG (Ikke motatt pr. 11.01.2023)

Oversendt liste over Hovedutstyr Tag fra Jon VAL: Det krever vedlikehold når AC4xx kontrollere oppgraderes til AC8xx da endres Severity nr. på alarmer og Eventer

Oversendt liste over Hovedutstyr Tag fra Eivind Hognestad ULA.

Det er brukt en blanding av understrek/Bindestrek på siste ledd

Skal vi legge in filter på «ACT»?

Utfordring:

Når jeg sjekker Ula eventer finner jeg ikke de som jeg får opp i CDF (Jupiter)

**Regler ULA vedr. «kall» fra SAP (Vidar Terjesen)**

* Ula Hovedutstyrsliste brukes som input «main\_components\_ula\_abb\_tags»
* Resultat etter spørring ref. Hovedutst. Liste havner i «found\_assets\_ula.csv»
* Kan vi bygge videre på «Kall» som ble brukt for Alarmer? Ja
* Bruk Severity 400-450 (401 til 409 er brukt i dag)
* tag\_states,(etterfulgt \_/-) :
  + 01: 'START/STOP OR OPEN/CLOSE'
  + 02: 'START OR OPEN (ON)'
  + 03: 'STOP OR CLOSE (OFF)'
  + 04: 'ESD/TRIP/SHUTDOWN'
  + 05: 'PULSED START OR OPEN (ON)'
  + 06: 'PULSED STOP OR CLOSE (OFF)'
  + 08: 'INHIBIT/DISABLE STOP OR CLOSE'
  + 14: 'F&G TRIP/SHUTDOWN'
  + 24: 'PSD TRIP SHUTDOWN'
  + 32: 'STOPPED OR CLOSED'
  + 34: 'COMMON ALARM/FAULT'
  + 36: 'TRIPPED (INTERNAL FAULT)'
  + 39: 'RUNNING/STOPPED, OPEN/CLOSED (1=RUNNING,OPEN)'
  + 40: 'FAULT (INTERNAL PROTECTION)'
  + 41: 'FAULT (INTERNAL PROTECTION)'
  + 42: 'FAULT (INTERNAL PROTECTION)'
  + 43: 'FAULT (INTERNAL PROTECTION)'
  + 44: 'FAULT (INTERNAL PROTECTION)'
  + 45: FAULT (INTERNAL PROTECTION)
  + START
  + STOP
  + FAIL

Dette blir duplisert mot VAL også

Sjekk:

25.01.23 AH: Flere Eventer fra Ula kommer dobbelt opp med samme dato fra CDF. Pål F. sjekker dette

## BlockLog presentasjon i CrewLog

Operatørene offshore ønsker at rapporten kun skal vise blokkeringer som er av interesse for handover mellom personell – altså manuelt satte blokkeringer.

Blocklog blir generert ulikt på de forskjellige installasjonene offshore

## Ivar Aasen

BlockLog fra IAA blir eksportert til .PDF daglig og lagret på D:\SOGODigital\ReportExportPdf\ offshore, deretter transportert til SAP Content Server – her blir den tilgjengeliggjort for SAP CrewLog app

NB: Blokkeringer utført i perioden 06:00/18:00 til 06:30/18:30 kommer ikke med i pdf fil

Genereringen av Block Log i Sogo blir kjørt mot PI server og tar ca. 30 min

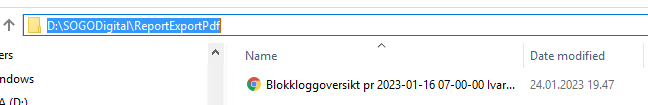
Transport morgen handover: **06:35**

Transport kvelds handover: **18:35**

Plassering blir som vist på bildet:

Server: VM-W-IMS006

Plassering D:\SOGODigital\ReportExportPdf\



Link SOGO: [Blocked and suppressed (akerbp.com)](http://sogo.akerbp.com/SOGO_test/#/index/bos?startTime=1482570540type%3Dall&showActiveOnly=true&tagName=&description=&fbTypeName=)

Utfordringer etter 25.03.2023 “Release”

Sjekket med IAA kontrollrom 29.03.2023 kl. 20:00 om BlockLogg var riktig (Dato/Tid):

* Den hadde riktig dato men klokkeslett var 05:59 og kom med to blocklogg filer
  + Oppdatering 12.04.2023 AH Dette er fikset
* Mange duplikate BlockLogger (Dette er det laget sak på hos Siemens)
  + Oppdatering 12.04.2023 AH Dette er fikset

Graphical user interface, application

Description automatically generated with medium confidence

Er det mulig å starte genereringen av BlockLog pdf fra SOGO manuelt på et hvilket som helst tidspunkt?

Hvis du tenker på «SOGO websiden» så er den kun «read only». Vi må legge dette manuelt inn i skriptet, eller lage en ny «manuel» kjøring skript som kjøres manuelt med å logge inn på server.

Ivar Aasen har bedt om å få blokkloggene inn i CrewLog til kl. 06:30 / 18:30, altså noe tidligere enn det vi har lagt opp til i systemet i dag.

* Dette involverer at Siemens med SOGO må starte jobben sin 5-10 min tidligere enn oppsettet er i dag
* Hva skjer da ift. lese-fil-rutine, da må det for IAA leses tidligere enn for de andre assetene (f.eks. 06:25 / 18:25)

## Ula

BlockLog fra ULA blir eksportert til .PDF daglig og lagret på [\\ULAPA52\BlockLog Reports\](file://ULAPA52/BlockLog%20Reports/) offshore, deretter transportert til SAP Content Server – her blir den tilgjengeliggjort for SAP CrewLog app

Transport morgen handover: **06:45 (endret 22.08.23)**

Transport kvelds handover: **18:45 (endret 22.08.23)**

Utfordringer etter 25.03.2023 “Release”

Sjekket med ULA kontrollrom 29.03.2023 kl.19:30 om BlockLogg var riktig (Dato/Tid): Den hadde riktig dato og klokkeslett, men kom med 2 like blocklogg filer.

* + Oppdatering 12.04.2023 AH Dette er fikset
* 24.04.23 AH: Bedt Barnabas Bitter om å endre tidspunkt til 06:20/18:20 =OK.
* 13.08.23 AH: Bedt Barnabas Bitter om å endre tidspunkt til 06:45/18:45 =OK.

Ønske fra ULA:

En knapp som kan generere BlockLog pdf. Ved behov

## Valhall

BlockLog fra VAL blir eksportert til .PDF daglig og lagret på [\\VALAP52\BlockLog Reports\](file://VALAP52/BlockLog%20Reports/) offshore, deretter transportert til SAP Content Server – her blir den tilgjengeliggjort for SAP CrewLog app

Transport morgen handover: **06:35**

Transport kvelds handover: **18:35**

Utfordringer etter 25.03.2023 “Release”

Sjekket med VAL kontrollrom v. Roger Holter 29.03.2023 kl.19:15 om BlockLogg var riktig (Dato/Tid): Den hadde riktig dato og klokkeslett, men kom med 2 like blocklogg filer.

* + Oppdatering 12.04.2023 AH Dette er fikset
* 24.04.23 AH: Bedt Barnabas Bitter om å endre tidspunkt til 06:20/18:20.

## Skarv

BlockLog fra SKA blir eksportert til .PDF daglig og lagret på SC21-P1-APP078 onshore K-IMS, deretter transportert til SAP Content Server – her blir den tilgjengeliggjort for SAP CrewLog app

Transport morgen handover: **06:35**

Transport kvelds handover: **18:35**

Utfordringer etter 25.03.2023 “Release”

Vi har ønske om å endre tidspunkt for logg/handover her på Skarv, fra 0645/1845 til 0630/1830.

Dette fordi vi normalt har handover på disse tidspunktene og vi ser da at Blokk logg ikke oppdateres før etter at vi er ferdige.

24.04.23 AH: Bedt Kristian Risnes om å endre tidspunkt til 06:20/18:20.

Sjekket med Skarv kontrollrom 29.03.2023 kl.19:37 om BlockLogg var riktig (Dato/Tid):

Den hadde riktig dato og klokkeslett PM 06:35 og kom med en blocklogg

* Jeg har også snakket med Ken-Roger og de oppdaget i går at noe som var blokkert og ble tatt inn igjen ila. skiftet ikke stod som to linker, men kun en; at den var blokkert. Og da gir ikke blokkloggen rett oversikt
* Det blir vedlagt flere blokklogger av samme versjon i skifthandover.
* Ser i tillegg ut til at feil "versjon" av blokklogg blir vedlagt.
* For skifthandover opprettet 07:02 26.03 og ferdigstilt 18:59 26.03 ligger blokklogg vedlagt som er generert 07:35 26.03. Vedlagt 2 ganger.
* For skifthandover opprettet 18:34 25.03 og ferdigstilt 07:01 26.03 ligger blokklogg vedlagt som er generert 06:35 25.03. Vedlagt 4 ganger.
* Siste status på aktive blokkeringer blir altså ikke lagt ved skifthandover.

## Alvheim

BlockLog fra SKA blir eksportert til .PDF daglig og lagret på C21-P1-APP036 onshore K-IMS, deretter transportert til SAP Content Server – her blir den tilgjengeliggjort for SAP CrewLog app

Transport morgen handover: **06:35**

Transport kvelds handover: **18:35**

Utfordringer etter 25.03.2023 “Release”

Sjekket med Avheim «Mathias» kontrollrom 29.03.2023 kl.19:50 om BlockLogg var riktig (Dato/Tid):

Den hadde riktig dato og klokkeslett 17:35 og kom med to blocklogg filer.

Det er feil BlockLog liste presentert fra Ny IMS (Alvheim bruker fortsatt gammel IMS), dette resulterer i at det ikke kommer opp kommentarer i BlockLog pdf (CrewLog) og det er tydligvis ikke riktig filter i bruk på ny IMS, da det ikke samsvarer med BlockLog liste fra gammel IMS.

24.04.23 AH: Bedt Kristian Risnes om å endre tidspunkt til 06:20/18:20 samt sette filter opp likt som gammel IMS.

* BlokkLogg kommentarer mangler (Blir siste versjon av IMS brukt?
* Viser automatiske blokkeringer som ikke skal være med?

## Edvard Grieg

Håkon Halvorsen Skibeli

vi er midt i migering til et nytt system, så litt vanskelig å se på nå. Lurer på om vi må komme tilbake om noen uker. F.eks innen 1. oktober. Vil det fungere? Helt kort så lurer jeg litt på om PDF egentlig er det dere ønsker. Er det ikke bedre om dere hadde fått tilgang på data-en? Skulle fint klart å få opp en PDF altså, men det virker på meg som litt baklengs. Vi prøver ogte å gå bort fra PDF-er.

Det skal ikke være noe problem i hvert fall. Jeg ser for meg noe i baner av 8-16 timers arbeid, men må som sagt komme tilbake med litt nærmere info

PDF er en løsning vi har

Vi kunne nok tilbudt et view, for å aksessere, så kunne noen av disse flinke SAP konsulentene ordnet med overføring. Eventuelt kunne kommet i en form for XML, CSV, Excel fomat.

APP122 er der jeg tror det vil ligge, men trenger ikke å ta det for god fisk. Mulig det er APP123, eller SQL031

## Veien videre for å bringe verdi fra CDF til CrewLog (Second opinion)

Pål Fister som leder etablering av Aker BP Digital grunnmur og Arvid Halrynjo Sr. Automasjons Engineer i CrewLog teamet har gått gjennom resultatene etter analysearbeidet av muligheter for å hente automatisk genererte hendelser og Alarmer fra CDF. Analysen av Alarmer og Eventer er utført mot VAL og ULA.

Vi har vurdert nytteverdien av dette slik:

* **Driftstatus/alarmer fra Hovedutstyr.** Det lar seg gjøre å utføre spørringer mot CDF hvor man isolerer spørringen mot Installasjon/hoved-utstyr/alarm aktiv/Event aktiv osv. man vil da kunne få en spørring presentert i CrewLog appen (Logg oversikt) som viser om en alarm eller event har vært aktiv den siste definerte tidsperioden (f.eks. 12 timers skift). SKR operatør kan da velge om denne alarmen eller Eventen er av verdi for neste skift og kan enkelt løftes over fra «logg» til «Oppgave» i CrewLog app.
  + Dette bør optimaliseres videre for å gi mer verdi og mindre alarmer i «Logg», det er da nødvendig å ha et API (application programming interface) for å behandle Alarm-eventene.
    - Alarm aktiv-kvittert-normal = ikke med i CrewLog
    - Alarm aktiv-kvitert = med i Crewlog

**Second opinion Pål Fister:**

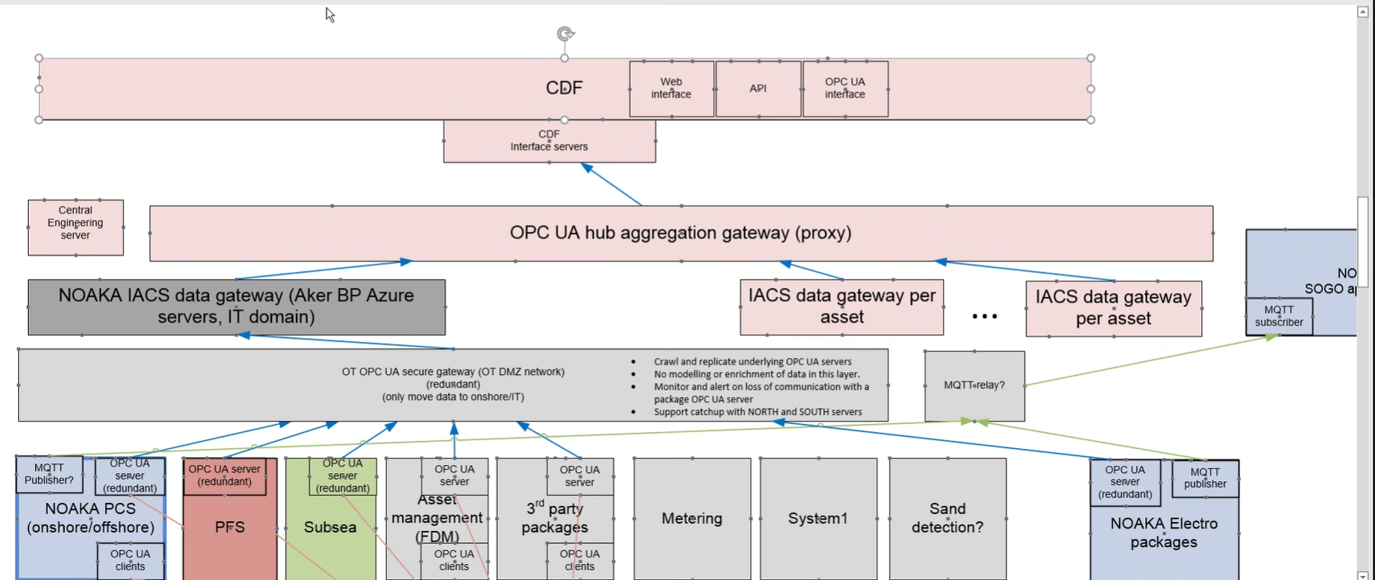
* + Det er nok til hjelp for operatørene å ha en oversikt på de alarmene som har vært aktivert innenfor siste shift, hvor operatøren velger videre om det er aktuelt å ta videre i handover
  + Enig i at dette kan optimaliseres videre for å gi mer konkrete alarmer ved at en ser på eventene/newState i hver alarm. Dette bør jobbes videre med og utvikles i CrewLog applikasjonen.
  + Har det egentlig noen verdi for handover å vise disse «isolerte» alarmene? Kunne det heller vært gjort en gjennomgang av de stående alarmene i aktiv alarmliste på OS?
* **Avvik fra kjøreplan stort redundant utstyr med alarmering i CrewLog**. Vi kan hente drift-status fra redundant utstyr. Slike kjøreplaner er pr. dags dato veldig manuelle i Aker BP, og består gjerne bare av et Excel-ark som operatørene har rutiner for å holde oppdatert.
  + Ref.: Kjøreplaner og -status for hovedutstyr. Hvis vi har kunnskap om nåværende kjørestatus for utstyret, og i tillegg informasjon om kjøreplanen til det respektive utstyret, så kan vi lage en veldig enkel logikk for å f.eks. varsle om feil. det er da nødvendig å ha et område/lag for å se på aktuell kjøreplan og status fra redundant utstyr

**Second opinion Pål Fister:**

* + - * Lar det seg gjøre å vedlikeholde kjøreplan i for eksempel CrewLog app? Ser for meg at dette kan bli umulig og at operatører mister tillit til CrewLog hvis alarmer/eventer ikke stemmer med virkeligheten.
      * Kjøreplan må digitaliseres og gjøres likt for alle installasjoner. Denne må kobles mot applikasjon som alarmerer til CrewLog ved avvik.
      * Det er viktig at kjøreplan ligger utenfor CrewLog og at dette ikke må vedlikeholdes i CrewLog

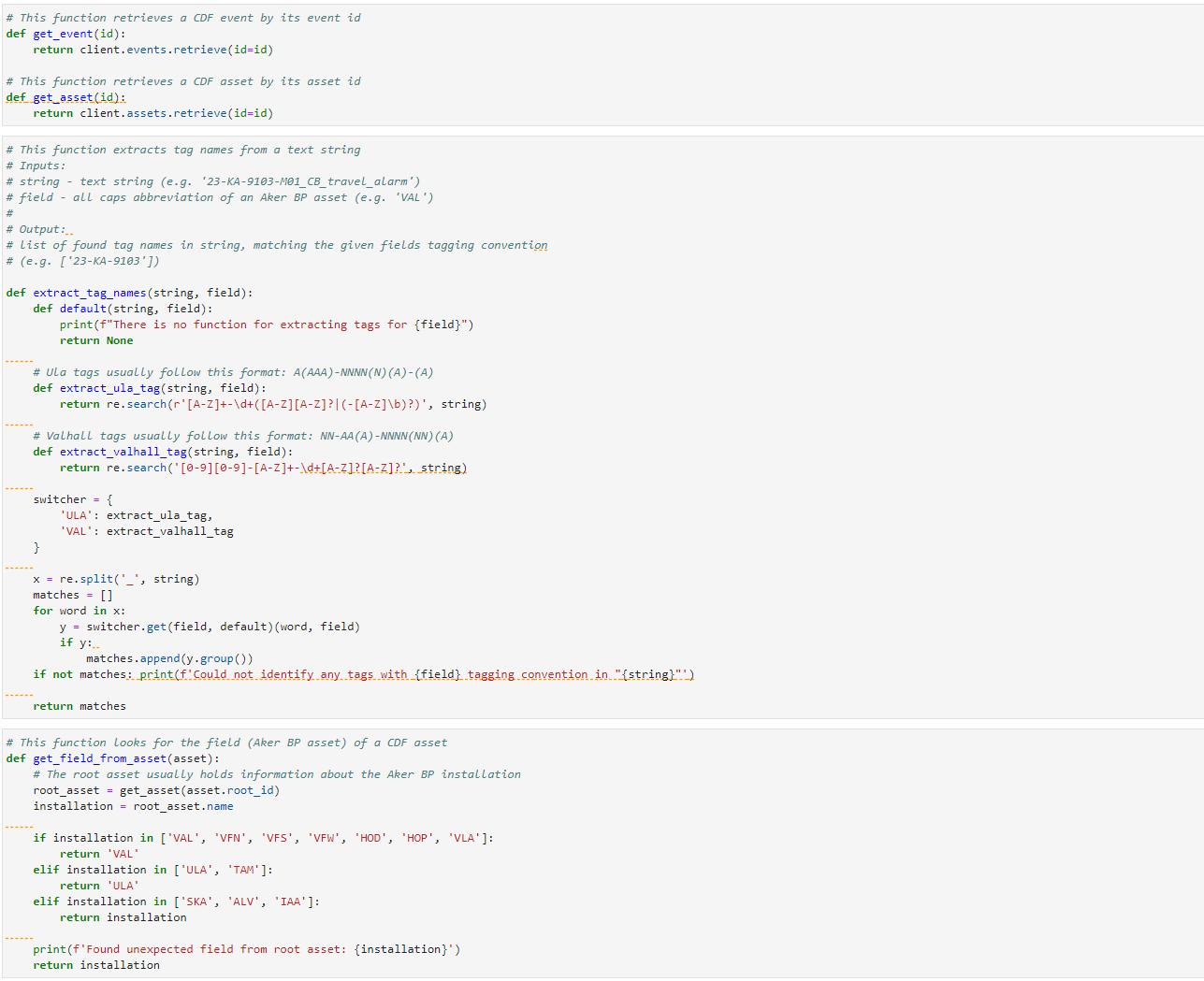
**Second opinion Ivar Andreassen møte fredag 4. november:**

* Hvis målet er å finne Root cause til en hendelse og presentere denne i CrewLog vil dette være omfattende og vil ta tid, dette må bli et eget prosjekt.
* Anbefaler tett oppfølging mot Cognite – Øystein Aspøy eller Torben Kristensen
* Ivar anbefaler direkte «kall» mot OPC UA/OT HUB (Flex modell), men var enig i at vi bør fortsette å hente Alarm og Events fra CDF slik vi gjør i dag.
  + Svar fra Pål: Strategien for Aker BP er at data skal hentes fra CDF. OPC UA HUB samler inn data og setter det i system før det pushes over til CDF, data kan hentes fra OPC UA HUB, men slik skal det ikke gjøres.

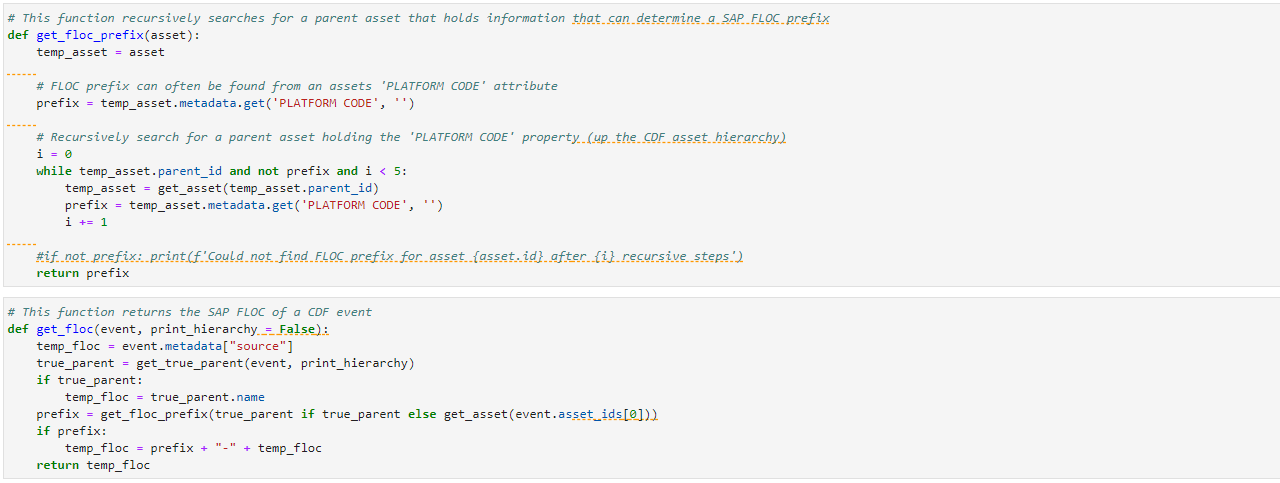


# Vedlegg

## Python funksjoner for å finne FLOC fra events kildetag







# Stakeholders

* Ula: Eivind Hognestad: [eivind.hognestad@akerbp.com](mailto:eivind.hognestad@akerbp.com)
* Valhall: Jon Fosshaug: [jon.fosshaug@akerbp.com](mailto:jon.fosshaug@akerbp.com)
* Skarv: Ken-Roger Olsen: [ken-roger.olsen@akerbp.com](mailto:ken-roger.olsen@akerbp.com)
* Alvheim: Mathias Moe Nilsen: [mathias.moe.nilsen@akerbp.com](mailto:mathias.moe.nilsen@akerbp.com)
* Ivar Aasen: Thomas Dragseth: [thomas.dragseth@akerbp.com](mailto:thomas.dragseth@akerbp.com)

I tillegg har vi med

* NUI: Marius Furumo: [marius.furumo@akerbp.com](mailto:marius.furumo@akerbp.com)
* Skarv (I tillegg til Ken-Roger): Hilde Kristin Brevik: [hilde.brevik@akerbp.com](mailto:hilde.brevik@akerbp.com)

Kobling Jupiter Lab:

[JupyterLab (cogniteapp.com)](https://akerbp.dshub.cogniteapp.com/user/arvid%20halrynjo/lab?redirects=1)

Gir lesetilgang til spesifikk datasett.

CDF Team må opprette tdatasett i hvert env

Git-repo hvor koden liggen

* Opprett git-repo i AkerBP gruppa på Git

Vanlig praksis:

* Sharepoint – CDF projects – lag egen mappe med info om eget prosjekt i word docs