

Digiriigi tulevikuarhitektuuri suunad

Aastal 2020 uuris Majandus- ja Kommunikatsiooniministeerium erinevatelt IT majadelt hetkeseisu erinevate uue generatsiooni tehnoloogiate ja arhitektuurilahenduste kasutamiste ja praktikate kohta. Nii küsimused kui ka vastused leiti olevat ebaühtlased ning otsustati teema võtta lahti peaarhitektide töögrupis, et ühiselt selgitada välja see, mis on erinevate suundade taga päriselt peidus ja mida erinevate sõnade all mõeldakse. 24.05.2021 käis peaarhitektide töögrupp sellel eesmärgil koos.

Käesolev dokument annab kiire ja selge ülevaate digiriigi erinevate haldusalade peaarhitektide poolt kokkulepitud ja läbi räägitud tehnoloogilistest suundadest järgmise generatsiooni digiriigi teenustes.

Digiriigi tulevikuteenused...

- ... on kaasaegselt pilvekõlbulikud
- ... on ehitatud autonoomsete mikroteenuste põhimõtetele
- ... kasutavad orkestreerimis-mootoreid
- ... kasutavad sündmuspõhise arhitektuuri lahendusi

Iga suuna kohta on välja toodud selle nii tehniline, kui ka äriline selgitus. Dokument on mõeldud nii IT juhile, kui ka toote- ja teenusejuhile ning arendusmeeskondadele.

Majandus- ja Kommunikatsiooniministeeriumi Digiriigi Arengu Osakond seab eesmärgiks nimetatud suundi ja nende suunas liikumist hinnata regulaarselt digiriigi edasises arengus.

PEAARHITEKTIDE TÖÖGRUPP:

- Kristo Vaher - MKM
- Rainer Turner - RIA
- Priit Parmakson - RIA
- Tarmo Hanga - RIA
- Raimo Reiman - RIA
- Mart Järvi - SMIT
- Siim Vene - SMIT
- Sten Viljus - KeMIT
- Martin Õunap - TEHIK
- Kalle Pärtlas - RMIT
- Viljar Tina - RIK

Tulevikuarhitektuuri suund I

Digiriigi tuleviku tehnilised teenused on kaasaegselt pilvekõlbulikud

MIKS

Digiriik peab olema skaleeruv vastavalt vajadustele. Kaasaegsed pilveplatvormid võimaldavad maksta selle eest, mida su teenused tegelikult kasutavad ja samal ajal võimaldavad jagada ressursse teiste teenustega hetkedel, kui teenustel on koormuse vajadus madalam.

See tähendab, et näiteks Maksu- ja Tolliameti füüsilise isiku tuludeklaratsiooni deklareerimise teenus ei pea igapäevaselt kasutama ja hõivama ressursside vajadust sama suures mahu, kui füüsilise isiku tuludeklaratsiooni avamise päeval ja sellele järgnevatel nädalatel - mis on riigile kallis.

Hea näitena võib välja tuua ka Amazon'i edu pilveteenuste pakkumisel: Amazon teadis, et e-poodide suurimad koormusvajadused on pühadel, mille tarbeks oli vaja suures mahus taristut. Selline taristu on aga hästi kallis pidada läbi terve aasta, mistõttu Amazon tegi strateegilise otsuse tekitada sinna peale teistele kasutatava pilveteenuse, millest tänaseks on saanud üks hinnatumaid pilveteenuseid.

SEOTUD TEHNILISED OOTUSED

- **Rakendus ja selle komponendid on skriptiga paigaldatavad ning paigaldamine ja taastamine toimub automatiseeritava skripti käivitamisega.**
miinimum: ainult rakendused
eesmärk: rakendused, andmebaasid ja muud seotud komponendid
- **Rakendus koosneb mitmest paralleelsest instantsist.**
- **Rakendus on automaatselt skaleeritav ja vajadusel kõrgkaideldav kahe asukoha vahel.**
miinimum: kõik active/passive
parem: rakendus active/active ja andmebaas active/passive
eesmärk: kõik active/active
- **Ideaal - aga mitte eesmärk omaette - 12 faktori metodoloogia rakendustes.**
https://en.wikipedia.org/wiki/Twelve-Factor_App_methodology

TÄHELEPANEKUD

- Mikroteenused on (*enamasti*) pilvekõlbulikud, aga kõik, mis on pilvekõlbulik ei pruugi olla mikroteenus - ehk ka ärivajaduste tõttu ehitatud monoliit võib siiski olla pilvekõlbulik.

Tulevikuarhitektuuri suund II

Digiriigi tuleviku tehnilised teenused on ehitatud autonoomsete mikroteenuste põhimõtetele

MIKS

Digiriik peab olema ehitatud paindlikult, et selles oleks võimalik sisse viia muudatusi vastavalt äriprotsessi vajadustele, seadustele ning kodanike ja ettevõtete ootustele võimalikult kiiresti.

See aga tähendab, et tehniline arhitektuur peab olema ehitatud paindlikult, et oleks võimalik sisse viia äri vajadustest lähtuvaid muutuseid. Paindumatu ning tihti monoliitne tehniline arhitektuur hoiab tagasi ka sellest sõltuvat organisatsiooni.

Mikroteenused on arhitektuurimuster tehniliste teenuste ehitamiseks ning on mõeldud selleks, et organisatsioonil säiliks paindlikkus. Autonoomsus tähendab seda, et teenused on võimelised enda piirides toimima ka siis, kui nendega seotud teised teenused on ajutiselt - *aga ka jäädavalt* - maas või ümberehitamisel. Sarnaselt nagu on autonoomsus tähtis organisatsioonil terviklikult - *ehk inimese haigestumine ei tohiks tähendada terve organisatsiooni kinnikiilumist* - on see tähtis ka tehniliste teenuste puhul.

Mikroteenused võimaldavad ka paremat tehnoloogiainigratsiooni, ehk kasutusele saab võtta uusi tehnoloogiaid ilma tervet lahendust ümberehitamata. Lisaks võimaldavad mikroteenused ka digiriigis suuremat taaskasutust, sest üksikute domeenide - *näiteks TARA näitel riigis autentiimine* - on jagatav ja taaskasutatav teistele asutustele

SEOTUD TEHNILISED OOTUSED

- **Infosüsteem > Domeenid > Mikroteenused > Komponendid**
Infosüsteem teenindab erinevaid domeene/rolle organisatsioonis; Domeenide vajadusi katavad erinevaid funktsioone pakkuvad mikroteenused; Mikroteenused koosnevad ühest või mitmest tehnilisest komponendist (*sh omaenda andmebaas*)
- **Infosüsteemid on ehitatud mitmest koosvõimelisest autonoomsest - ehk väheste otseste sõltuvustega - mikroteenusest.** Mikroteenus on teenuse komponent, milles on kokku koondatud kõik funktsioonid, mis muutuvad samadel põhjustel - erinevatel põhjustel muutuvad komponendid on tõstetud eraldi mikroteenusteks.
- **Mikroteenused ei jaga teiste mikroteenustega andmebaasi, failisüsteemi, mälu jms.** Integratsioonid on ainult üle kokkulepitud teenusekihtide (API'd, sh sõnumiruumide API'd)
- **Äriloogilised tehnilised komponendid on kasutatavad erinevate kasutajaliideste poolt.**
Kasutajaliides ja teenuse funktsionaalsus on loogiliselt eristatud kihid ja suhtlevad üle API.

Digiriigi tuleviku tehnilised teenused **kasutavad orkestreerimis-mootoreid**

MIKS

Digiriigis on mitmed teenused äriliste seoste ja sõltuvustega mitte ainult üle erinevate haldusalade, aga ka haldusala siseselt üle mitme haldusala enda teenuse ja andmekogude. Mitme teenuse tegevuste orkestreerimine - ehk erinevate teenuste tegevuste toimumise tagamine selges ahelas ja selgetel tingimustel - on seetõttu tähtis.

Traditsiooniliselt ehitatakse sellist protsesside sõltuvusahelat infosüsteemide enda sisse, aga see tekitab raskuseid selle hilisemal muutmisel - näiteks olukordades, kus arenduspartner vahetub või on möödas piisav hulk aega ja tehakse tehnilisel muutmisel inimlikke vigu. See lõppeb tihti sellega, et sildistatakse hea lahendus enneaegselt *legacy*'ks ehk pärandvaraks.

Tihti on suurem äriprotsess ehitatud ja kirjeldatud mitte ainult ühe teenuse, vaid mitme teenuse - ka üle mitme haldusala teenuse - sisse, mis teeb eksponentsiaalselt keeruliseks äriprotsessi muutmise, kui ka sellest selge ülevaate saamise. Rahvusvaheliselt tuntud teenused nagu Netflix ja Spotify on seetõttu pidanud strateegiliselt tähtsaks suure äriarhitektuuri pildi orkestreerimist just läbi tehniliste orkestraatorite, mis annab kai juhtidele tegeliku visuaalse pildi - mitte tõlgenduse - äriprotsessist.

Visioonidokumendis *NEXT GENERATION DIGITAL GOVERNMENT ARCHITECTURE* tõstisime hüpoteesi, et kuigi orkestreerimislahendused ei pruugi olla ideaalne lahendus tehniliste teenuste äriarhitektuuriliste seoste haldamiseks, siis võib olla see täna just esmajärjekorras esimene lahendus, mida tõsisemalt uurida ja selle jätkusuutlikkust valideerida just selliste vajaduste ulatuses, nagu mitut organisatsiooni koosvõimelisena hõlmav ja teineteisest sõltuv digiriik.

SEOTUD TEHNILISED OOTUSED

- **Rakenduste enda sisse on kirjutatud ainult rakenduse enda piiridesse jääv äriloogika.**
Üle mitme teenuse sõltuv äriprotsess ei tohi olla kirjutatud teenuste enda sisse.
- **Kui tegu on teenusega, millel täna on või tekib tulevikus seos sündmusteenustega või muude haldusalade üleste äriprotsessidega, siis peab olema teenus ehitatud kujul, et ta on orkestreeritav.**
Ehk rakendusel peab olema API või rakendus peab oskama enda teenust pakkuda mõne sõnumiruumi kaudu.
- **Orkestraatori mootoreid nullist ise mitte ehitada.**
Katsetada rakendusi nagu Camunda, Flowable, jt avatud lähtekoodiga BPM lahendusi.

Digiriigi tuleviku tehnilised teenused kasutavad sündmuspõhise arhitektuuri lahendusi

MIKS

Sõltuvused osapoolte vahel - nii organisatsiooni enda inimestevahelised sõltuvused või siis tehniliste komponentide vahelised sõltuvused - on üks keerulisemaid kohti mis iganes protsessi disainis. Sarnaselt nagu igapäevaelus oleme frustreeritud, kui kodupanga veeb mingil põhjusel ei tööta, siis me ei taha, et elu sellest täiesti seisma jääb. Organisatsiooni teenuste eest vastutajatena ei taha me selliseid sõltuvusi tekitada ka enda teenuste ja nende tehniliste komponentide vahele.

Teenuste autonoomsuse tagamiseks on sündmuspõhise arhitektuuri lahendused, nagu sõnumiruumid - üks parimaid lahendusi. See tähendab, et üks-ühele sõltuvuste vahele tekitatakse ruum, mis on füüsilises maailmas võrreldav näiteks igapäevase avatud kontori või kabinetiga, kuhu saavad liituda erinevad osapooled ja seal vahetada omavahel erinevaid ülesandeid või infokilde. Samal ajal võimaldavad sellised tehnilised sõnumiruumid ka "jäarele jõuda", ehk ajutine ruumis mitteviibimine ei takista teenuse enda edasist tööd - ruumiga uuesti liitudes saab vahepeal kaotsiläinud sõnumid uuesti kätte.

Selline lähenemine võimaldab ka andmete liigutamist laiemalt, kui kahe osapoole vahel, ehk ühte sõnumit või päringut saab korraga edastada mitmele osapoolele ning ruumiga saab panna liituma uusi teenuseid ilma teisi teenuseid ümberehitamata. Selline paindlikkus on igapäevaelus organisatsioonide protsessi disainivatele inimestele iseenesestmõistetav, aga enamus infosüsteemid pole tehniliselt täna sellistel põhimõtetel ehitatud.

SEOTUD TEHNILISED OOTUSED

- **Rakenduste sisse mitte ehitada otsepäringuid teiste lõpp-infosüsteemide pihta, kui selleks pole selget põhjendatud vajadust.**
- **Rakenduse integratsioonid andmete saatmiseks või pärimiseks või ülesannete andmiseks disainida asünkroonselt sõnumiruumide kaudu - kui selleks pole põhjendatud vajadust teisiti.**
- **Sõnumiruumi lahendusi nullist ise mitte ehitada.**
Katsetada lahendusi nagu Kafka ja RabbitMQ.
- *Digiriigis on aga pigem mõistlik kasutada rumalaid sõnumiruumi - ehk ruum ise äriprotsessist midagi ei tea ning teadmine ja vastutus jääb andmekogude ja infosüsteemide enda kätte.*
- *Tulevikus on plaan tagada ka sellise lahenduse kasutamine hõlpsalt otse üle X-tee niiõelda X-ruum lahendusena. Täpsem ajakava ja plaan selle osas on alles selgumisel.*

Mida saad sina teha juhi või arhitektina VEEL teha, et tagada enda teenuste areng nendes suundades?

- **Tee ja arenda ja ehitä ise seda, mida sina tead paremini - muu osta teenusena sisse.** Ehk kui su organisatsioon ei oma eesmärki ehitada maailma parimat pilvetaristut, siis tasub pilveteenust osta sisse teenusena kelleltki teiselt. See võimaldab säästa aega, raha ning keskenduda enda organisatsiooni tugevustele ja ärivajadustele.
Kontrollküsimus: Kas see toode/teenus, mida me teeme ja arendame ja haldame, on meie unikaalne või teevad sama asja ka teised?
- **Organisatsioonis on ülimalt tähtis tagada vastutavates teenustetiimides CI/CD (Continuous Integration/Delivery) võimekus.** See tähendab, et arendusmeeskonnad peavad automatiseerima oma arendusprotsesside ja tarneprotsesside kõikvõimalikud rutiinid. Live keskkonda minek peaks olema nupuvajutusega lahendatud.
Kontrollküsimus: Kas sinu toode/teenus läheb Live'i ühe nupuvajutusega?
- **Kaalu kvaliteedi tagamiseks Chaos Engineering lähenemise juurutamist kas täielikult või vähemalt tähtsamatel teenustel.** See tähendab, et sinu teenuste tehniline taristu ei tohiks kaardimajana kõik maha variseda, kui üks või teine suvaline tehniline komponent oma töö ettearvamatult lõpetama peaks. Tehniline teenuste arhitektuur, mis suudab nii vastu pidada on märgatavalt autonoomsem.
Vt. veel: https://en.wikipedia.org/wiki/Chaos_engineering
Kontrollküsimus: Kui me ühe misiganes komponendi enda tootest/teenusest maha võtame, mis juhtub? Kas teised teenused jäävad püsti? Kuidas me saame kindlad olla? Kas meil on olemas selge plaan sellisteks olukordadeks?
- **Valideeri kasutatavate tehnoloogiate pikemat jätkusuutlikkust.** Iga tehnoloogia kasutamine - sealhulgas ka programmeerimiskeelte kasutamine - on ajas muutuv. Tagada tuleb, et ei kasutataks tehnoloogiaid, mida enam ei toetata (näiteks mis ei saa enam turvauuendusi) ning ei kasutataks programmeerimiskeeli, mille osas pole turul enam kompetentsi või mille osas on risk, et kompetents on hääbumas. Mõlemal juhul on vaja organisatsioonil tagada selge vastutusega 'exit plan' sellisest tehnoloogiast väljamigreerimiseks.
Vt. veel: <https://www.tiobe.com/tiobe-index/>
Kontrollküsimus: Kas meil on kasutuses tehnoloogiaid, mis ei ole TIOBE Top 25 indeksis? Top 50? Mida me teeme, et seda riski minimeerida?
- **On tähtis vältida sõltuvust ühte kindlasse partneritesse ja seotud tehnoloogiatesse.** Vastasel juhul tekib olukord, kus partner dikteerib üksinda hinda ning muutub raskemaks tehnoloogiamigratsioon.
Kontrollküsimus: Kas meie kasutatavad rakendused ja nende koodi on meil võimalik edasi arendada ka teiste partneritega? Kas meie rakendused on meie jaoks avatud lähtekoodi litsentsidega (MIT License, etc).
- **On tähtis tagada, et tehnilistel teenustel on olemas põhifunktsionaalsuse ulatuses API'd ja et neid saaks kasutada ilma traditsioonilise veebibrauseri kasutajaliidese või muu seotud rakendusest.** See on vajalik digiriigis nii sündmusteenuste orkestreerimistel, aga mitte ainult, sest see võimaldab enda teenuste funktsionaalsuseid kasutada ka enda teiste tiimide poolt või ka üle haldusalade.

Kontrollküsimus: Kas meie tehnilistel teenustel on olemas API'd ja kas kasutajaliidesed kasutavad ise ka neid samu API'sid?