Hvordan håndtere kompleksitet?

25. oktober 2013, Margunn Aanestad

Hvordan håndtere kompleksitet?

- Hva gjør bedrifter i praksis
  - Legacy-systemer
  - Datavarehus og 'data analytics'
  - Integrasjonsteknologier
  - IT-styring
  - Arkitektur-basert styring
- Begreper fra pensumlitteratur

Legacy-systemer

- Gammelt system/teknologi/applikasjon som fortsatt brukes, for eksempel fordi:
  - … det inneholder viktige og verdifulle data
    - Ofte brukes ordet ”heritage-system” for å vektlegge positiv verdi
  - …det fungerer helt greit
  - …det er for dyrt å endre det
  - …det er for risikabelt å gjøre noe, det må holdes i drift 24/7
- Hva er problemet med gamle systemer?
  - Manglende dokumentasjon/kjennskap til systemet
  - Ufleksibelt: man kan for eksempel ikke legge til nye datafelt, nye behov og bruksområder, effektivisere bedriftens arbeidsprosesser osv.
  - Dyrt å drifte, vanskelig å overhold lovkrav og lignende.

Hvordan leve med legacy-systemer?

- Endringsstrategier:

  Drastiske (”rip and replace”) eller evolusjonære (”enhance and evolve”)

  - Migrering – gradvis overgang til nye systemer, og utfasing av gamle (ETL)
  - Wrapping – ”bygge inne” systemene (f.eks. grensesnitt)

- Software-arkeologi, reverse engineering

- Code refactoring – rydde opp:
  - Restrukturere kodens ’indre’ struktur uten å endre ’ytre’ oppførsel
  - Martin Fowler: “Refactoring: Improving the Design of Existing Code”

Datavarehus

- En løsning som ofte brukes for å samle, konsolidere og tilgjengeliggjøre informasjon fra ulike kildesystemer
  - Sammenstille informasjon for tverrgående analyser (på tvers av støttesystemer), dvs. for beslutningsstøtte heller enn produksjon/operasjons-støtte
  - Man tar underlagsmaterialet for gitt, dvs. søker ikke redesign av kildesystemene men jobber med det som er
  - Bottom-up (data marts) eller top-down (modell-drevet)
- Komponenter:
  - Operasjonelle databaser (for eksempel ERP-system)
  - Datatilgang: ETL-verktøy (Extract, Transform, Load)
  - Metadata: Data Dictionary
  - Informasjonstilgang: BI-verktøy (Business Intelligence)
  - (”Beyond BI” – ”Big Data” – ”Data Analytics”)

Håndtering av heterogenitet

- ”Informasjons-infrastrukturer er heterogene”

- En samling av ulike IT-systemer kan være heterogen på ulike måter:
  - Dataformatet (syntaktisk heterogenitet)
  - Struktur i datamodell (strukturell heterogenitet)
  - Betydningen av data (semantisk heterogenitet)
  - System-heterogenitet, f.eks. hardware, OS

- Semantisk ulikhet = samme begrep i ulike systemer har ulikt meningsinnhold

Integrasjon – et vidt begrep

- Hva menes egentlig med integrasjon?
  - 'Grad' av integrasjon: Kommunikasjon, interoperabilitet eller integrasjon?
  - Hva slags "sammenkobling"? Tett/løs, permanent/tidsavgrenset, enveis/toveis/flerveis, synkron/asynkron osv.
- Hvilket nivå?
  - Teknisk (dataformat, syntaks).
  - Semantisk (meningsinnhold).
  - Organisatorisk (rolle i prosess, sekvens)
- For hvem, fra hvilket perspektiv, for hvilket behov?
  - Ulikhet mellom de ulike casene:
    - MCC (Maritime Classification Company)
    - Hydro (Bridge & SAP)
    - NorthOil
    - Rikshospitalet

Fra ad hoc til "top-down" integrasjon:

- I praksis har integrasjon ofte vært ad-hoc, dvs. situasjonsbestemt og skreddersydd
- Bevegelse mot mer generiske tilnærminger (standardiserte og/eller styrte, sentraliserte) og fra tette til løsere koblinger
- Historisk:
  - (Cut and Paste-mulighet)
  - Remote Procedure Call (RPC)
    - Med mer: Remote Method Invocation (RMI), Common Object Request Broker Architecture (CORBA), Microsoft DCOM (Distributed Component Object Model), .NET Remoting, Java Remote Method Invocation (RMI), RPC-style Web services.

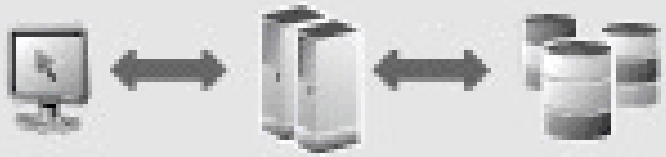
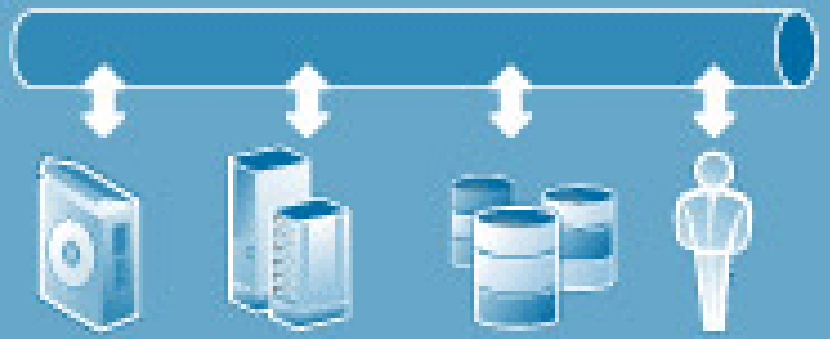
Et historisk blikk: Fra ad hoc til "top-down" integrasjon:

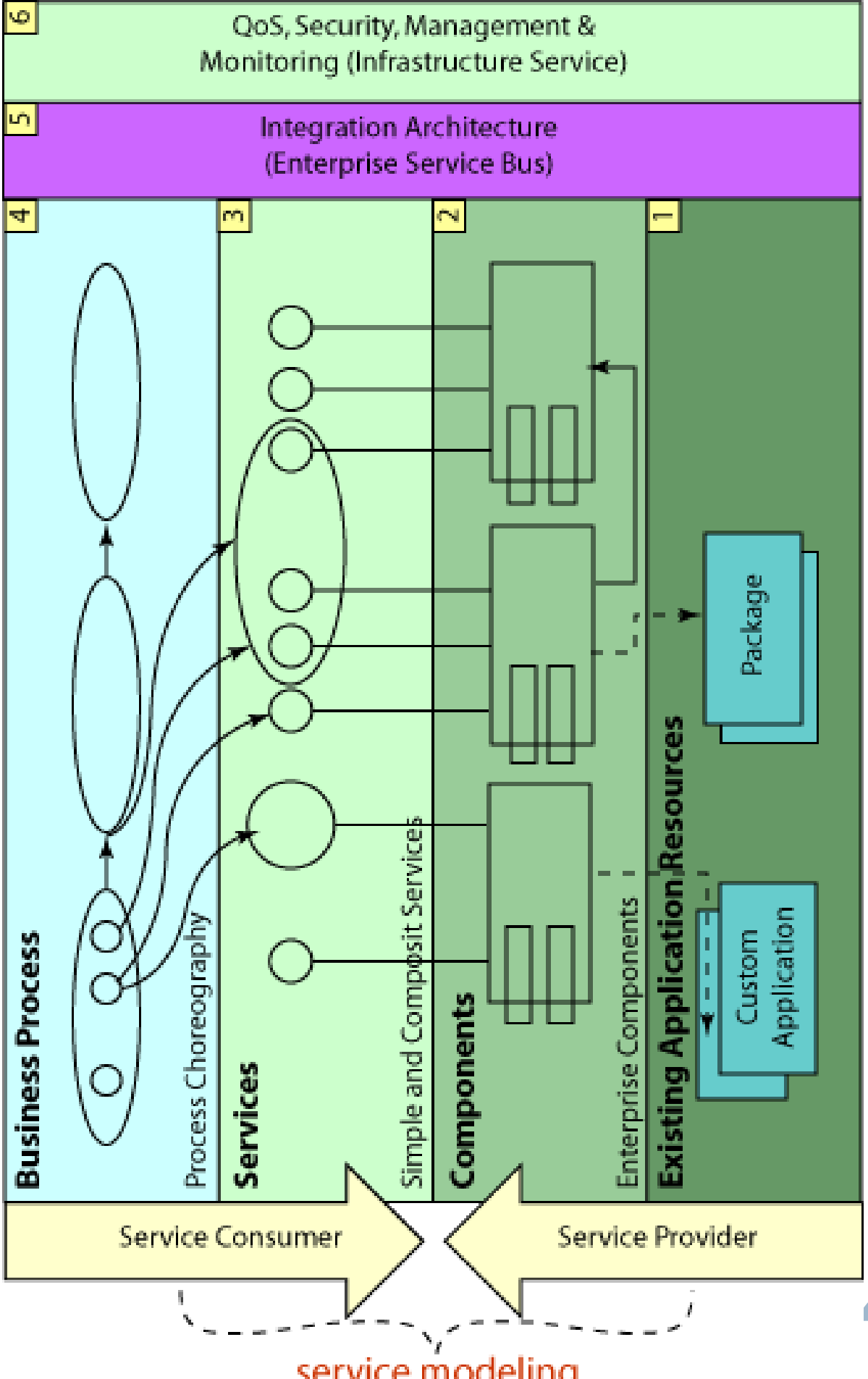
- Noen begreper:
  - Meldingsbaserte løsninger
    - EDI, XML og ebXML
  - "Middleware"
    - Software laget for å koble sammen andre systemer (dvs. brukes mellom ulike 'silo-systemer')
  - EAI (Enterprise Application Integration)
  - Buss-basert (Enterprise Service Bus)
    - "interaksjons-standard"
- Fra ad hoc integrasjon (reaktiv) til arkitektur-baserte tilnærminger (top-down, modell-drevne, pro-aktive)

Service-Oriented Architecture

- Tjeneste-orientert arkitektur
- Tjenester = veldefinert funksjonalitet samlet i software-komponenter. Disse tjenestene beskrives ('annonseres') og kan brukes av andre

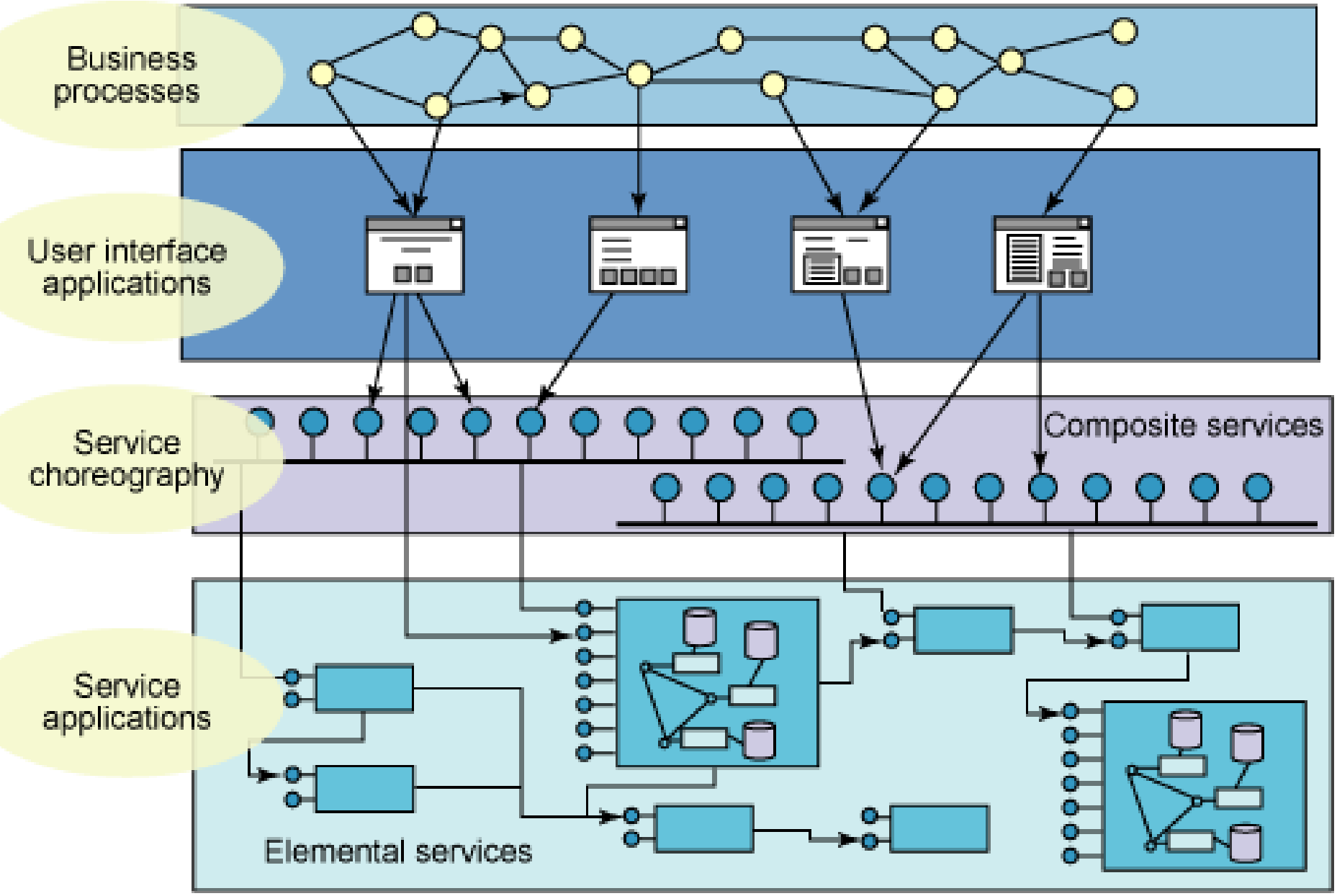
- OASIS' definisjon:
  - "A paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations."
    - (OASIS, the Organization for the Advancement of Structured Information Standards 2006)

3-Tier Architecture	SOA
	
Homogenous	Heterogeneous
Language Dependent	Language Independent
Centralized Application Tiers	Massively Distibuted Services
Code Centric Applications	Flexible Composite Applications
Request/Reply Driven	Request/Reply, Pub/Sub, Events
HTML Pages	AJAX Rich Internet Applications


6
QoS, Security, Management &
Monitoring (Infrastructure Service)
5
Integration Architecture
(Enterprise Service Bus)
4
3
2
1
Business Process
Process Choreography
Services
Simple and Composit Services
Components
Enterprise Components
Existing Application Resources
Package
Custom Application
Service Consumer
Service Provider
service modeling

Prinsipper bak SOA

- En samling tjenester (i praksis ofte Web Services) som kan brukes av flere (gjenbruk av tjenester er poenget)
- Koblinger mellom tjenestetilbyder og tjenestesøker/bruker etableres ved behov (tilgjengelige tjenester beskrevet i tjenestekatalog)
- Det forhandles via et standardisert tjenestegrensesnitt (og dette resulterer i en tjenestekontrakt) heller enn gjennom API'er
- Innkapsling og modularisering (skille grensesnittet/kommunikasjon fra hvordan funksjonaliteten er implementert) (dvs. språk og plattform-uavhengig)
- Tjeneste-orkestrering/tjenestekomponering sentralt


Business processes
User interface applications
Service choreography
Composite services
Service applications
Elemental services

Service-Oriented Architecture

- Storebrand tidlig ute:
  - *Se gjennomgang av deres erfaringer på:*
  - *forum.dataforeningen.no/attachment.php?attachmentid=1877*

- NAV's nye pensjonssystem er basert på SOA, skal gradvis migrere over alle andre systemer på samme plattform
  - http://www.idg.no/nyheter/article98286.ece

- Spesialisthelsetjenesten i Norge
  - http://www.nasjonalikt.no/Publikasjoner/Tjenesteorientert_arkitektur_i_spesialisthelsetjenesten_hovedrapport_full_v1_0e.pdf

- Verdiskaping fra muligheten for storskala integrasjon vs. kostnadsreduksjon/effektivisering fra gjenbruk

Sentrale SOA (Web Services) standarder:

- Web services bygger på tre XML-baserte standardser:
  - SOAP (Simple Object Access Protocol) standardiserer meldingsformatet ("konvolutt" og beskrivelser av innholdet). SOAP-meldinger kan utveksles over ulike protokoller
  - WSDL (Web Services Description Language) XML-dokument som beskriver tjenestens grensesnitt mot "utenverdenen". (Hva den gjør og hvordan lage en SOAP-melding som kan kommunisere med den)
  - UDDI (Universal Description, Discovery and Integration protocol) gir oversikt over tilgjengelige web-tjenester, deres egenskaper, lokasjon og krav.

IT governance

- Bred definisjon: "… the leadership and organisational structures and processes that ensure that the organisation's IT sustains and extends the organisation's strategies and objectives." [IT governance institute]
- Smalere definisjon: "Specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT.“ [Weill & Ross 2004]
  - Sentraliserte, desentraliserte eller føderterte modeller for fordeling av ansvar for IT-relaterte beslutninger
  - Skille mellom ansvar for ulike nivåer, strategiske beslutninger, forretningsprosesser, applikasjoner, informasjon, infrastruktur.

Motivasjon for sterkere styring av IT

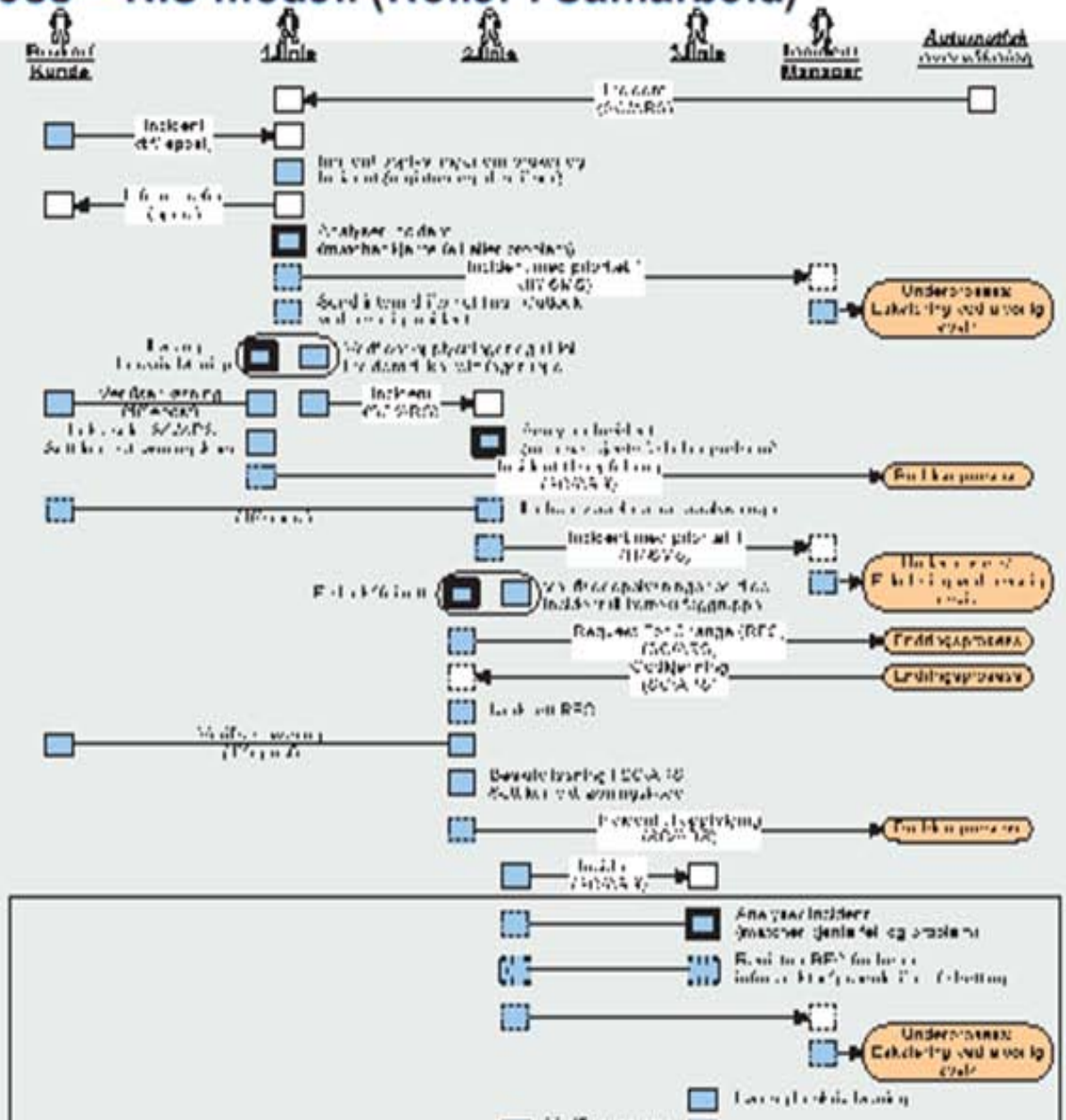
- Økt avhengighet av IT og dermed større sårbarhet
- Økt kompleksitet som medfører økte kostnader til vedlikehold
- Økt behov for at IT-løsninger 'spiller sammen'
- Økte krav om kvalitet på IT-tjenester (f.eks. sikkerhet)

- Noen utbredte rammeverk for styring av IT:
  - ITIL (Information Technology Infrastructure Library) (drift)
  - CoBIT (Control Objectives for Information and Related Technology)
  - Prince2 (Projects in Controlled Environments, version 2)
  - KVU+KS1+KS2 (prosjektstyring i norsk off.sektor)

ITIL (Information Technology Infrastructure Library)

- Strukturert rammeverk av "best practices" for håndtering av IT i en organisasjon
  - Dekker it-drift, support, infrastruktur, systemforvaltning, sikkerhet m.m.
  - Omfatter begreper, beskrivelser av prosesser, og prosedyrer, sjekklister m.m.
    - Norsk ITIL terminologiliste 2009: www.itsmf.no
  - Standardisering av infrastruktur, av prosesser og ytelse (kvalitet, sikkerhet)
  - Prosess-modell (PDCA) (slektskap med ISO9000 og CMM/CMMI)
    - .. "etablere sammenhengende arbeidsprosesser, klare roller og entydige ansvarsplasseringer"

Incident prosess – RIS modell (Roller I Samarbeid)



Erfaringer med ITIL

- Primær gevinst: det settes fokus på prosessene, man får gjennomført strukturering og prosess-forbedring
- Bedre samarbeid med kunde, enklere kvalitetsovervåkning, økt prosesseierskap (deltakelse og klargjøring av roller), nytt styringssystem…
- Utfordringer: Organisatorisk endringsprosjekt, grensesnitt mellom prosessene osv…
- Klassisk konflikt mellom 'rigid' drifts-seksjon og 'fleksibel' utviklingsavdeling

ITIL® V3 PROCESS MODEL

Long Term Business Objectives

THE BUSINESS

SERVICE STRATEGY

- Definition of Business Service Requirements
- Determination of Market Space, IT Policies and Strategies
- Specification of Service Portfolio
- Demand Management
- Financial Management

Business Requirements (Utility)

- Service Pipeline
- Service Catalogue
- Retired Services

EXTERNAL SUPPLIERS

Underpinning Contracts (UC)

Service Metrics Recommendations

SERVICE OPERATION

FUNCTIONS

- Service Desk
- Operations Management
  - Operations Control
  - Facilities Management
- Technical Management
- Applications Management Lifecycle

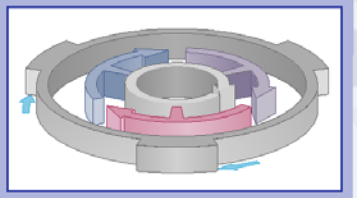
Development

PROCESSES

- Event Management
- Incident Management
- Request Fulfilment
- Problem Management
- Access Management

Process & Technical Metrics

CONTINUAL SERVICE IMPROVEMENT



- Seven-Step Improvement Process
- Deming Cycle & CSI Model

Process Metrics

SERVICE DESIGN

- Service Level Management
  - Service Catalogue Management
  - Supplier Management
- Availability Management
- ITSCM
- Capacity Management
- Information Security Management

Process Metrics

Alerts, Warnings (Integrated Tool)

SKMS

- CMIS
- AMIS
- SCD

CMS

- CMDB's
- KEDB

- DML

SERVICE TRANSITION

- Change Management
- Service Asset and Configuration Management
- Knowledge Management
- Release and Deployment Management

Service Transition Package (Release)

Service Design Package (Warranty)

CHANGE ADVISORY BOARD

- Requests for Change Appraisal
- Approval
- Schedule

Abbreviations:

SKMS – Service Knowledge Management System
CMS – Configuration Management System
CMDB – Configuration Management Database
AMIS – Availability Management Information System
CMIS – Capacity Management Information System
KEDB – Known Error Database
SCD – Supplier Contract Database
ITSCM – IT Service Continuity Management

ILX GROUP

For further details please contact us at:
W: www.ilxgroup.com T: +44 (0)1270 611600
F: +44 (0)1270 628513 E: sales@ilxgroup.com

ILX Group plc. Registered in England & Wales No. 03525870.
Registered Office: One London Wall, London, EC2Y 5AB
ISBN 978-0-9544884-4-4 Published Nov 2007.

Arkitektur-basert tilnærming

- Sentrale begreper innen arkitektur-tilnærminger:
  - Man jobber med flere **dimensjoner**
    - Perspektiver eller 'views'
  - Starter med å **kartlegge** inputs, føringer, betingelser
    - Slike forutsetninger kan for eksempel være forretningsstrategier og prioriteringer, teknologi-relaterte beslutninger eller føringer, brukerkrav fra forretningssiden, eksisterende IS, eksisterende arkitektur. Samt ikke-funksjonelle krav relatert til tilgjengelighet, pålitelighet, skalerberhet, sikkerhet, ytelse, interoperabilitet, modifiserbarhet, vedlikeholdbarhet, brukbarhet og håndterbarhet.
  - Må **balansere** ulike hensyn:
    - Trade-offs og kompromisser står sentralt i arbeidet

Hva slags arkitektur?

- Informasjonsarkitektur
  - Fra forretningsobjekter til logisk datamodell, hvordan strukturere (hvilke databaser etc.)
- Applikasjonsarkitektur
  - Definerer hvilke applikasjoner og integrasjoner som skal understøtte arbeidsflyten (hvilke skal fases ut, hva skal inn)
- Tjenestearkitektur
  - Hvilke tjenester skal finnes, hva skal være felles, hva skal tilbys
- Teknologiarkitektur
  - Plan for bruk av teknologier på en måte som koordinerer utviklings/driftsressurser og konsoliderer nøkkelteknologier

… og andre begreper:

- Sikkerhetsarkitektur
- Forretnings-/virksomhetsarkitektur
- Teknisk arkitektur
- Systemarkitektur
- Softwarearkitektur
- Integrasjonsarkitektur
- …

Begrepet ”Views” er sentralt i arkitekturmodellen, søker å representere dette mangfoldet.

Noen arkitektur-rammeverk

- **The Zachman Framework** for Enterprise Architecture
- **4+1 View Model of Architecture** er et rammeverk for å modellere software arkitektur. Fire "viewpoints" (logisk, prosess, utvikling, fysisk) og et "Scenario view". (ikke fokus på videreutvikling eller enterprise-nivå)
- **Federal Enterprise Architecture Framework** (US Fed Govt). Fire nivåer, mest for arkitektur-planlegging.
- **Open Distributed Processing – Reference Model (RM-ODP)** Fra ISO og ITU-T: a Reference Model for Open Distributed Computing (RM-ODP). Software-arkitektur-fokus
- **TOGAF** The Open Group Architectural Framework (TOGAF). Enterprise architecture
- **DODAF** US Department of Defense Architecture Framework.

Zachman

- Zachmans rammeverk (ikke en metode, men rammeverk)
- Zachman (1987): ”A Framework for Information Systems Architecture”(IBM systems journal)
- Senere: ”Zachman’s Framework for Enterprise Architecture”
- Mye brukt, ofte inkorporert i andre rammeverk. Fokus på ulike “stakeholders” og deres perspektiver

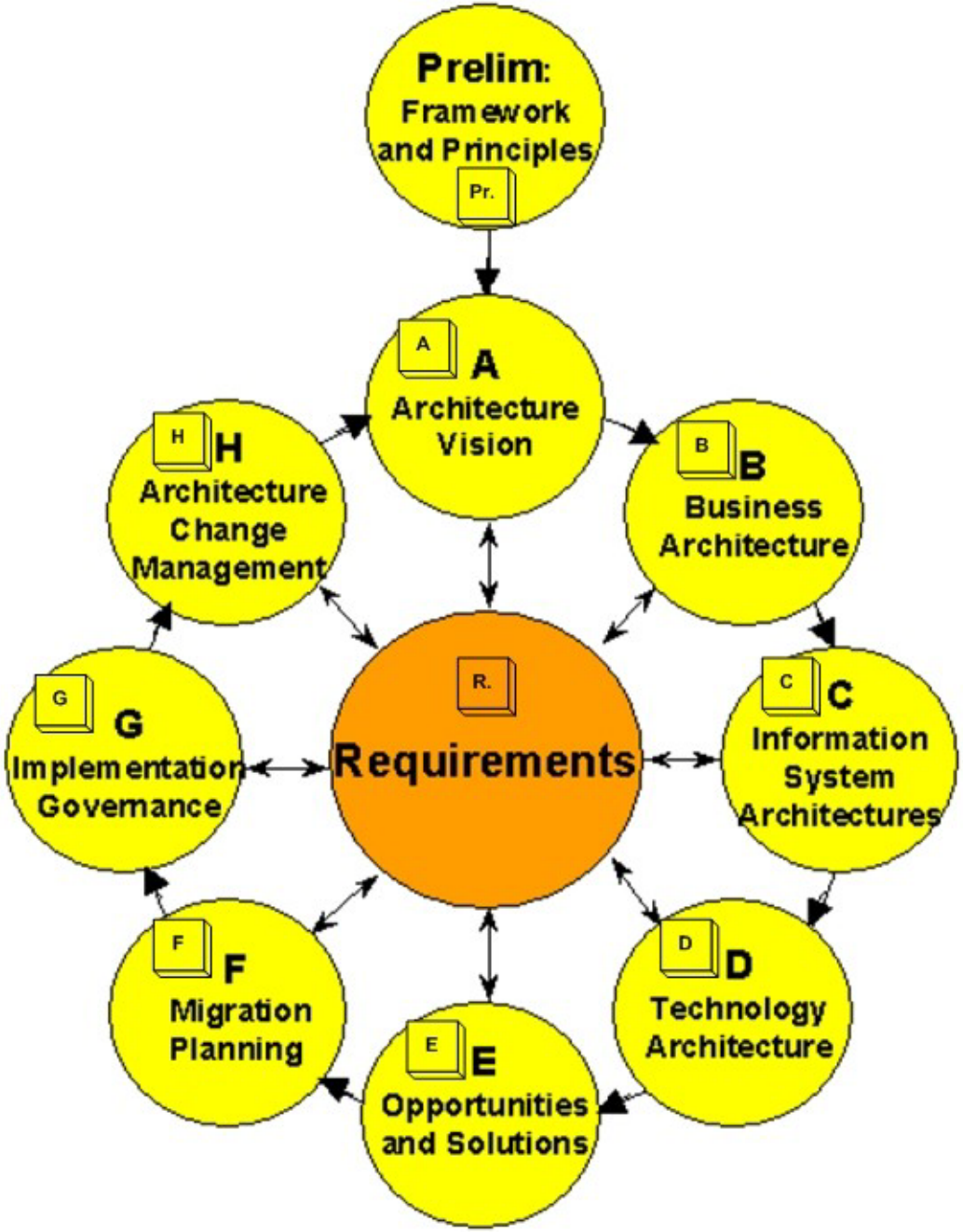
- http://www.zifa.com/

	D Data	F Function	N Network	P People	T Time	M Motive
1 Planner's View	Business Things Entity = Class of Business Thing	Processes Performed Function = Class of Business Process	Business Locations Node = Major Business Locations	Organizations People = Major Organizations	Significant Events Time = Major Business Event	Goals and Strategy Ends/Means = Major Business Goals
2 Owner's View	Semantic Model Ent = Business Entity Rel = Relationship	Process Model Proc = Process I/O = Resources	Logistics System Node = Location Link = Linkage	Work Flow Model People = Organization Work = Work Product	Master Schedule Time = Business Event Cycle = Business Cycle	Business Plan End = Objective Means = Strategy
3 Designer's View	Logical Data Model Ent = Data Entity Rel = Relationship	Application Architecture Proc = Function I/O = User Views	System Architecture Node = IS Function Link = Line Properties	Interface Architecture People = Role Work = Deliverable	Processing Structure Time = System Event Cycle = Processing	Business Rule Model End = Structure Means = Action
4 Builder's View	Physical Data Model Ent = Segment/Table Rel = Pointer/Key	System Design Proc = Function I/O = Data Elements	Technology Architecture Node = Hrdwr/Sftwr Link = Line Specs	Screen Architecture People = User Work = Screen Format	Control Structure Time = Execute Cycle = Component	Rule Design End = Condition Means = Action
5 Integrator's View	Data Definition Ent = Field Rel = Address	Program Proc = Statement I/O = Control Block	Network Architecture Node = Addresses Link = Protocols	Security Architecture People = Identity Work = Job	Timing Definition Time = Interrupt Cycle = Machine Cycle	Rule Design End = Sub-Condition Means = Step
6 User's View	Data Ent = Rel =	Function Proc = I/O =	Network Node = Link =	Organization People = Work =	Schedule Time = Cycle =	Strategy End = Means =

TOGAF (The Open Group Architecture Framework)

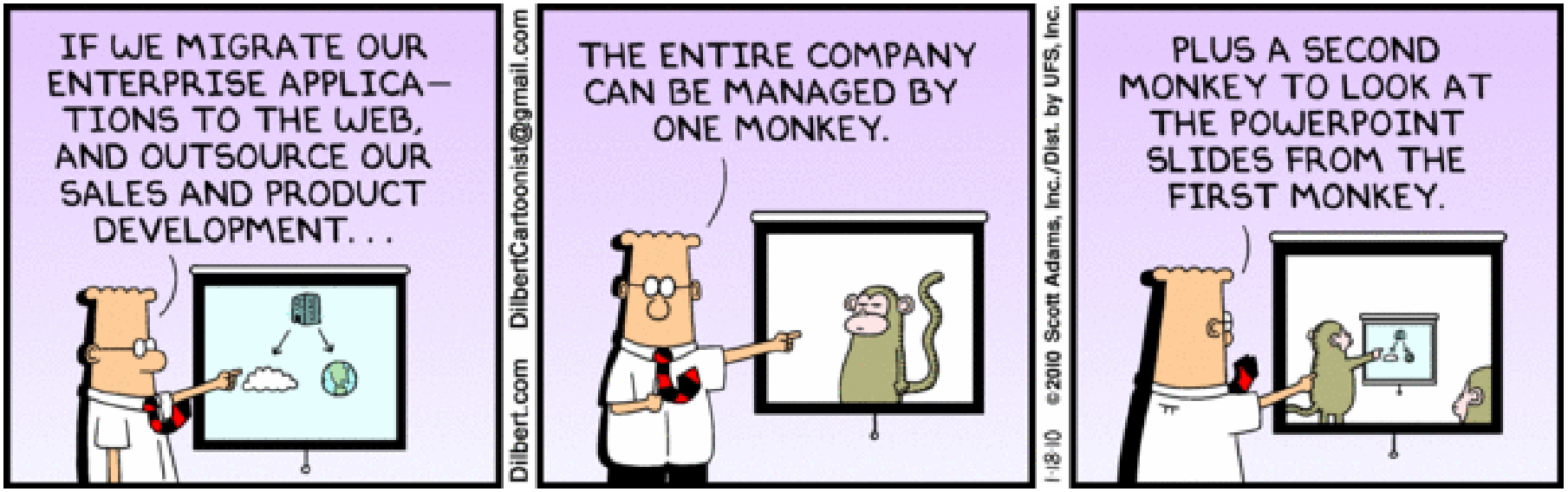
- The Open Group (www.opengroup.org) består av førende it-leverandører
- TOGAF er en metode og verktøy for å utvikle enterprise-arkitektur (inkl. design, planlegging, implementering og løpende håndtering/styring)

- http://www.opengroup.org/togaf/


Prelim:
Framework
and Principles
Pr.
A
Architecture
Vision
B
Business
Architecture
C
Information
System
Architectures
D
Technology
Architecture
E
Opportunities
and Solutions
F
Migration
Planning
G
Implementation
Governance
H
Architecture
Change
Management
R.
Requirements

Arkitektur i norsk off. sektor

- I forvaltningsmeldingen St.meld. nr. 19 (2008-2009) *"Ei forvaltning for demokrati og fellesskap"* presenteres de sju arkitekturprinsippene som skal følges:
  - Tjenesteorientering
  - Interoperabilitet
  - Tilgjengelighet
  - Sikkerhet
  - Åpenhet
  - Fleksibilitet
  - Skalerbarhet
- FAOS – Felles Arkitektur for Offentlig Sektor
- http://www.difi.no/filearchive/2009-10-08-arkitekturprinsipper-2.0.pdf
  - http://www.difi.no/ikt/it-arkitektur
- EU: European Interoperability Framework:
  - http://ec.europa.eu/idabc/en/document/7728
  - Pan-European Egovernment Services (PEGS)


IF WE MIGRATE OUR ENTERPRISE APPLICA-TIONS TO THE WEB, AND OUTSOURCE OUR SALES AND PRODUCT DEVELOPMENT...
THE ENTIRE COMPANY CAN BE MANAGED BY ONE MONKEY.
PLUS A SECOND MONKEY TO LOOK AT THE POWERPOINT SLIDES FROM THE FIRST MONKEY.
Dilbert.com DilbertCartoonist@gmail.com
1-8-10 ©2010 Scott Adams, Inc./Dist. by UFS, Inc.

Oppsummering: Styringstilnærminger

- Ulike strategier for å håndtere kompleksitet
  - Integrasjon
    - Reaktiv: problemløsning/opprydning i etterkant
  - Arkitektur-/modell-drevet utvikling
    - Proaktiv: Planlegging i forkant
    - Men: det er alltid en ”installed base”
    - Sitat: ”Arkitektur er prosess”
      - Hvordan nå målbildet? Må definere og gjennomføre en migrasjonsprosess

Styring av informasjonsinfrastrukturer

- Pensumartikler:
  - Hanseth og Lyytinen: Design-prinsipper
    - Hvordan forholde seg til dynamisk kompleksitet?
    - Case: Internetts fremvekst
  - Hanseth og Aanestad: 'Bootstrapping'
    - Oppstartsproblematikk, hvordan nå 'kritisk masse'?
    - Case fra norsk helsesektor
  - Braa m.fl., : 'Fleksible standarder'
    - Hvordan lage standarder som passer i mange ulike kontekster og som kan endres ved behov?
    - Case: HISP – definisjoner av standarder for datasett
  - Nielsen og Aanestad: 'Control Devolution'
    - Proprietær kontroll eller åpen infrastruktur?
    - Case: infrastruktur for sms i Norge (Telenor/Netcom)

Hanseth og Lyytinen (2010)

- Hanseth og Lyytinen sier:
  - Før har II-forskere gjort deskriptive (beskrivende) case-studier, ofte av problematiske II-prosesser
  - Nå trenges en **design-teori** for informasjonsinfrastrukturer – hvordan skal man bygge dem?
  - De formulerer 5 designprinsipper (og 19 'regler'), basert på Complex Adaptive Systems-teori
  - Internetts historie: et case å lære av/illustrere prinsippene

- Kjerneutfordring: å håndtere dynamisk kompleksitet
  - Hvordan bygge noe hvis man ikke aner hva det skal bli, hvem som skal bruke det, eller hva kravene vil være?

- Kort historisk oversikt
  - 1969: De 4 første ARPANET-nodene ble koblet sammen
  - 1972: e-post tatt i bruk for at utviklerne skulle kommunisere
  - 1973: internasjonale noder (London og Kjeller, Norge)
  - Fra Arpanet til Internet
    - TCP/IP-protokollen tatt i bruk ca 1982/3
    - Det ble et større nettverk (men fortsatt for forskere) rundt 1985
      - Oslonett (IFI, 1991) første åpne webserver utenfor universitets-miljø (privatpersoner, småbedrifter)
    - Domain Name System (1983)
    - World Wide Web (1991, CERN)
    - … osv.

Hva er spesielt med Internetts utvikling?

- Brukere = utviklere
  - Dvs. insentiver og innsats var tett koblet
- Gradvis teknologisk utvikling
  - Først basal teknologi for fildeling og meldings-utveksling
  - Med internetts vekst kom nye utfordringer (for eksempel behov for DNS, IPv6, nye organisatoriske former osv.)
- Åpen arkitektur
  - Åpent for heterogene nettverk (Inter-networking), –> kunne "spise" sine konkurrenter)
  - Ende-til-ende arkitektur (utvikling delegert til brukere/'markedet')
- Anne standardiserings-strategi (enn telekom)
  - Krav om "running code" før løsningen ble standard

Internett-standarder

- Slagord: (the IETF Credo)
  - ”We reject: kings, presidents and voting. We belive in: rough consensus and running code”
- Internett-standarder kalles ”RFC’er” (RFC+nummer)
  - ”Request For Comments”

- Hva skiller Internett-standardene fra vanlige standarder?
  - Læringsorientert, erfaringsbasert, iterativ prosess
  - Fleksibilitet gjennom flere nivåer av ”modenhet” på standardene og ulike grader av ”tvang” (Required – recommended – elective – limited use – not recommended)

Hanseth og Lyytinen: To sentrale design-utfordringer

- ”Bootstrappings”-utfordringer (Oppstart)
  - Skal man lykkes i å etablere noe som helst, må det gi en viss verdi til de første som skal ta det i bruk
  - Hvordan gjør man det når verdien er avhengig av mange brukere? (nettverks/kommunikasjonsteknologier)

- Vekst/utviklings/tilpasnings-utfordringer
  - Satser man bare på ad hoc design og lokale løsninger, vil man støte på problemer (stagnasjon). Design må forholde seg til fremtidig endring i både skala og funksjonalitet
  - Hvordan gjør man det når man ikke vet hva som vil skje?

Design-prinsipper:

- For bootstrappings-problemet:
  - 1. Design initially for usefulness
  - 2. Draw upon existing installed base
  - 3. Expand installed base by persuasive tactics

- For adaptiv vekst-problemet:
  - 4. Make each IT capability simple
  - 5. Modularize the II by building separately its principal functions and sub-infrastructures using layering and gateways

- Design principles for the bootstrap problem:
  - 1. Design initially for usefulness
  - 2. Draw upon existing installed base
  - 3. Expand installed base by persuasive tactics

Prinsipp 1:
Man hadde store visjoner/scenarier, men laget først enkle løsninger (fjern-innlogging, filoverføring, epost) som hadde direkte bruksverdi for utviklerne selv

Prinsipp 2:
- TCP/IP kunne kjøre på ulike underliggende nettverksløsninger (radio, satellitt, datanettverk, modem over telefonlinjer..)
TCP/IP bundlet med UNIX BSD
Web: ikke bare html-dokumenter, men kunne embedde andre data (feks fra databaser) i html ved hjelp av CGI – økte bruksverdien dramatisk

- Design principles for the adaptability problem:
  - 4. Make each IT capability simple
  - 5. Modularize the II by building separately its principal functions and sub-infrastructures using layering and gateways

Til prinsipp 4:
'Simplicity' var et uttalt krav til løsningene (i den første RFC'en)
Minimale protokoller – liten risiko for feil/tvetydighet i implementasjon

Til prinsipp 5:
Enkle arkitektur-prinsipper (for eksempel ende-til-ende-tenkning, modularisering)
Lagdeling: transport-, tjeneste- og applikasjons-infrastruktur
-Åpnet for innovasjon 'oppå' TCP/IP
-Muliggjorde distribuert håndtering for eksempel W3C for webteknologi
Koble parallelle II ved hjelp av gateways (transisjon IPv4 til IPv6)

- ’Kultivering av installert base’ versus mer tradisjonelle utviklingsmodeller:
- Vannfallsmodellen/fossefallsmodellen
  - lineær og sekvensiell: grundig analyse og spesifisering før design og implementering
  - Modifisert: iterativ - gjentatte ’runder’ med vannfallsmetodikk
- Boehms spiralmodell
  - Orientert mot risiko-minimering
- Iterative/Inkrementelle modeller, evolusjonære og agile modeller

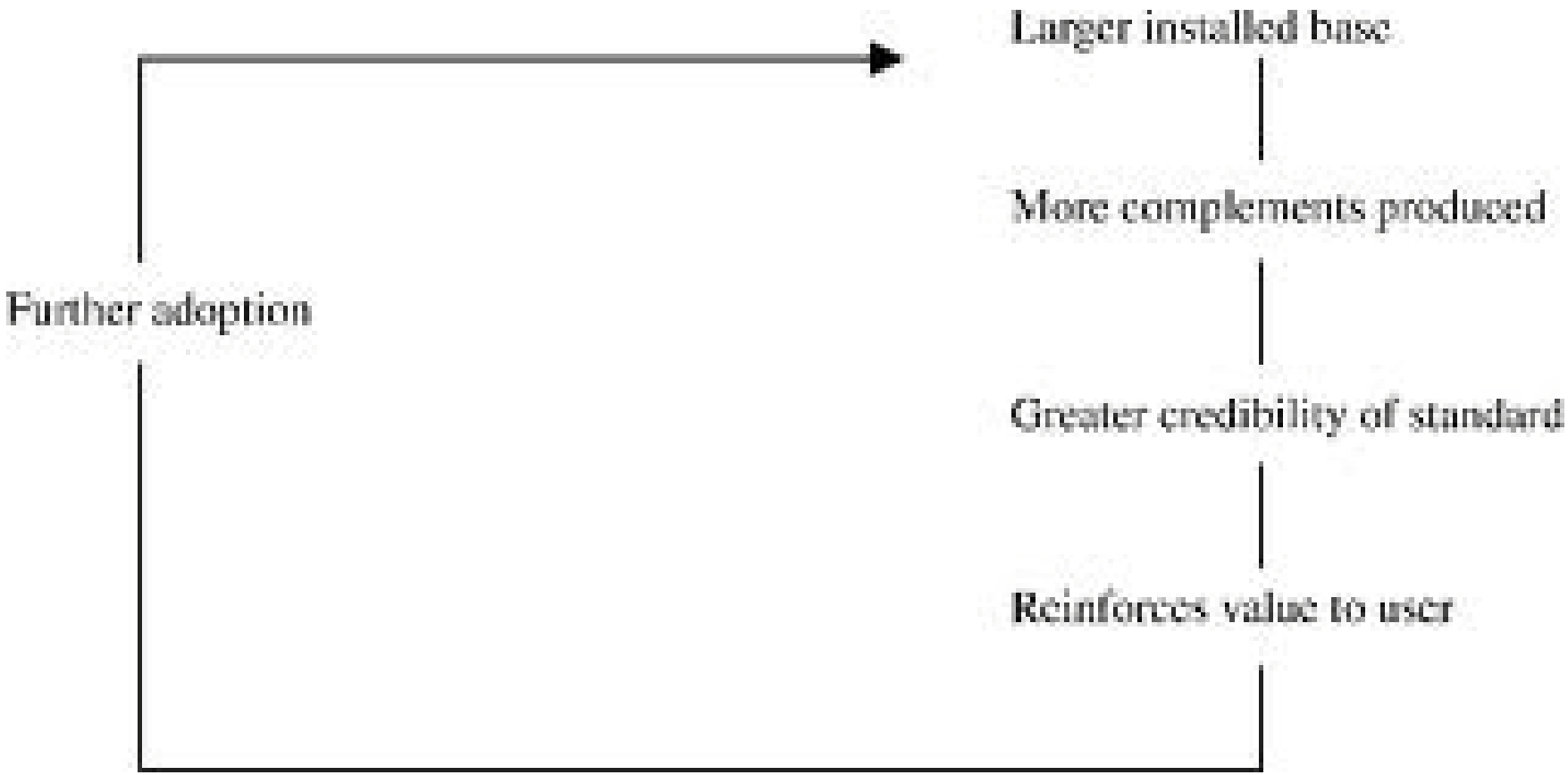
Kontroll eller fleksibilitet?

- Fundamental spenning:
- Planlegging og estimering av ressursbruk
  - Hvordan får man startet et prosjekt om man ikke kan dokumentere dette?
- Behov for endringer og justeringer underveis
  - Hvordan får man til dette dersom man har faste planer og budsjett?

Er kultiveringsstrategier realistiske?

Nettverkseffekter

- Klassisk (ressurs-basert) vs. Informasjons-økonomi
- Kommunikasjonsteknologier: verdien av teknologien avhenger av nettverkets størrelse
  - Verdien er ikke bare avhengig av teknologien isolert sett, men av hvor mange andre (og hvem) som har den samme (dvs. kommunikasjons-partnere)
- Eksternaliteter (negative/positive)
  - Eksempel: komplementøre produkter
- Selvforsterkende mekanismer:
  - 'Sterkere', mer attraktivt produkt/standard - Selvforsterkende mekanismer - Momentum, komplementære produkter osv.
- Derfor viktig prinsipp: styrk installert base, øk antall brukere



Hvordan få satt en slik prosess i gang?

Bootstrapping

- Et spesial-tilfelle av kultiverings-strategi:
- En strategi for oppstarts-utfordringer – hvordan oppnår man "kritisk masse"…
  - når man ikke har makt til å pålegge bruk,
  - og ei heller penger til å subsidiere brukerne
- En strategi for bruker-innrullering
- I dag: et av tre case i artikkelen (Hanseth og Aanestad 2003)

”Bootstrapping” – hvordan starte opp noe nytt?

- Velg ”start-punkt”:
  - Alle bruksområde er ikke like komplekse, kritiske osv.
  - Alle tekniske løsninger er ikke like (pris, kompleksitet)
  - Noen brukere er mer motiverte enn andre
- Poeng: Utnytt disse forskjellene
- Bruk eksemplets makt: ’det virker jo!’
- Bygg en plattform som kan være utgangspunkt for videre utbredelse

En 'Bootstrapping'-strategi

- Start med den enkleste og billigste løsningen
- Velg motiverte og kunnskapsrike brukere
- Velg bruksområde som er enkle og ikke-kritiske
- Utvid heller enn å bytte ut
- Velg bruksområder der gevinsten er umiddelbar og lett synlig
- Velg bruk/teknologi som har et stort potensiale for å få mange brukere

..

- Bruk dette så langt som mulig, dra inn flest mulig brukere
- Bruk samme løsning på mer innovative måter som gir mer gevinst
- Bruk løsningen for mer kritiske oppgaver
- Bruk løsningen for mer komplekse oppgaver
- Vurder å forandre løsningen slik at nye oppgaver kan løses
- (Gjenta fra begynnelsen).

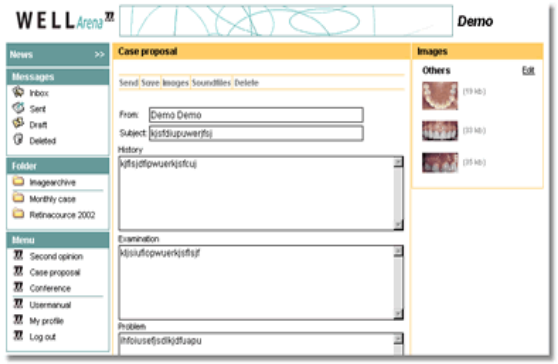
Bootstrapping-ideen:

- Bruk det du har, ikke gjør deg avhengig av det du ikke har

- ”Deliver a better today, rather than promise a better tomorrow”.
  - (fra Karl Weicks historie om Adam Michnick og ”Solidaritet” i Polen)
  - Andre begreper: Tidlige gevinster (quick wins), lavthengende frukter
  - Redusere kompleksitet, redusere avhengighet av ressurser, redusere risiko for katastrofale feil, maksimere læring….

Telemedisin

"Telemedisin er: Undersøkelse, overvåkning, behandling og administrasjon av pasienter og personale via systemer som gir umiddelbar tilgang til ekspertise og pasientinformasjon uavhengig av hvor pasienten eller relevant informasjon er geografisk plassert."



Telemedisin

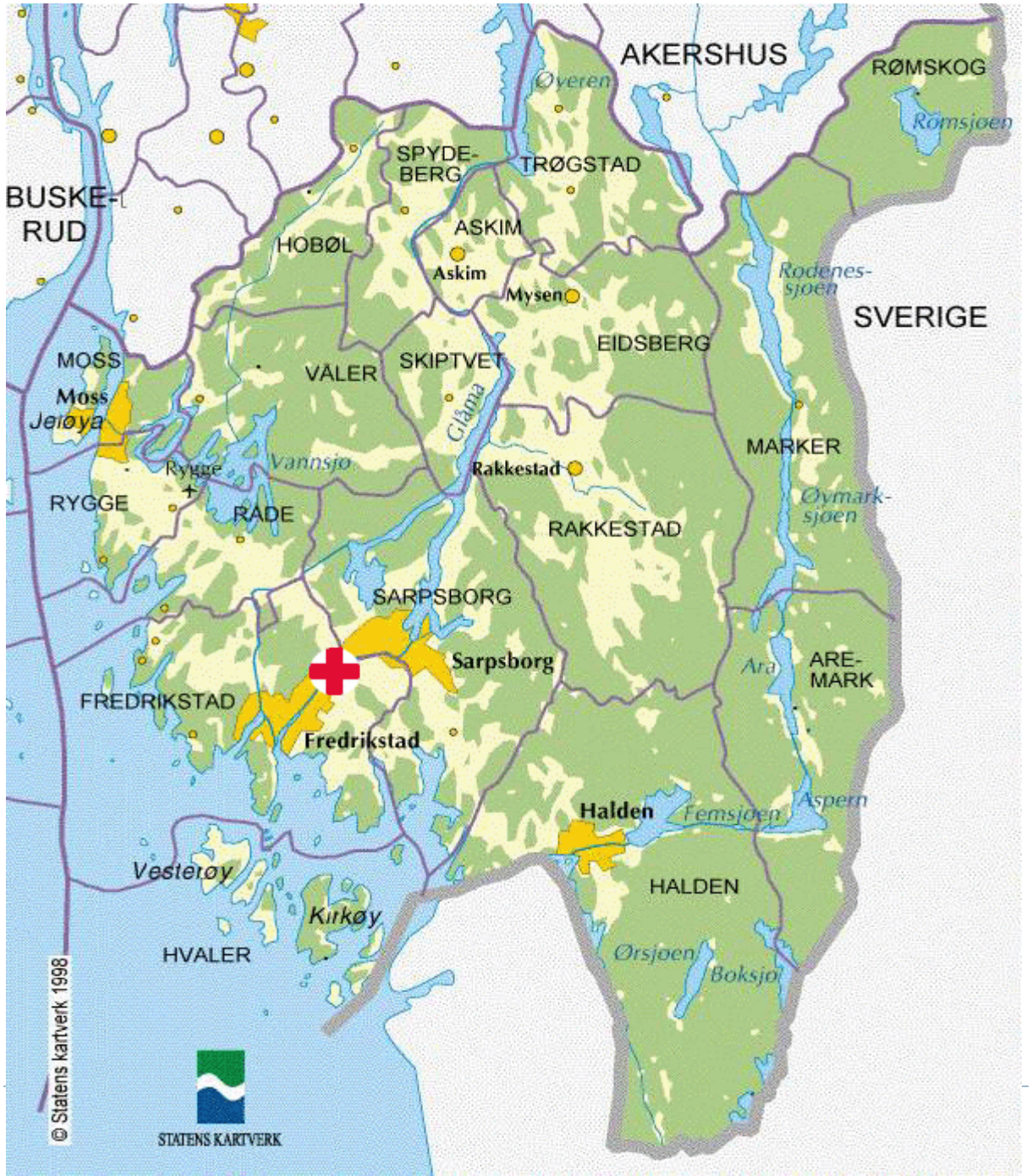
- Mange (pilot-)prosjekter, men fortsatt lite rutinebruk
- Utfordring: For å oppnå positive effekter (f.eks. innsparing av reiseutgifter) kreves endring hos flere aktører og deler av helsevesenet samtidig
  - Timeplanen hos sender og mottaker må stemme (vakt?)
  - Juridisk – hvem har ansvar for medisinske beslutninger?
  - Hvem skal betale? Sykehus, kommune, legekontor? (takster?)
- 'Ond sirkel':
  - Dvs. telemedisin trenger bred oppslutning for å lykkes, men får ikke en slik oppslutning før disse effektene kan tallfestes/bevises
  - Effektene blir ikke realisert før man har denne støtten og oppslutningen

Østfold fylke

250 000 innb.
3600 kv.km.

Fram til 1998:
5 sykehus
5 akuttmottak

Etter 1998:
2 akuttmottak
(Moss og Fr.stad)



Hjerteinfarkt

- Infarkt: Deler av hjertemuskelen mister blodforsyningen pga. en blodpropp. Kan være mer eller mindre alvorlig.

- 'The golden hour': behandling innen en time
  - Trombolyse (medikamentell behandling, kontraindikasjonar)
  - Utblokking/stenting av årene vha. kateter/PTCA

- Tidsforsinkelser:
  - Fra pasient blir dårlig til ambulanse kommer
  - Kjøretid
  - Fra ankomst (akuttmottak) til behandling starter (hjerteavdeling): 'door-to-needle time'
  - Total 'call-to-needle time' er avgjørende

MobiMed-teknologien

- Sendere i ambulanser (GSM-basert)
- EKG kan overføres til mottaker på sykehus
- Lege tolker oversendt EKG
- Tidlig deteksjon av hjerteinfarkt, og deretter:
  - Trombolytisk behandling før innleggelse
  - Og/eller: Omgå akuttmottaket og bringe pasienten direkte til hjerteintensiven

MobiMed-historien (1)

- 1997: Lege + ambulansesjåfør besøkte Falun (Sverige) for å se MobiMed i bruk
  - Søkte fylkeskommunen om støtte, fikk nei
- Februar 1998: Mottaker ved hjerteavdelinga i Fredrikstad, sender i to Halden-ambulanser
  - Utstyr lånt fra leverandør (ingen økonomisk støtte)
  - Frivillig deltakelse frå legene på hjerteavdelingen
  - Formål: tidlig diagnose, omgå akuttmottaket
- Prosjektet kunne dokumentere tid spart (ved å omgå akuttmottaket) for 16 pasienter i 1998.

Case fra Nina Mikkelsons hovedoppgave (2002)

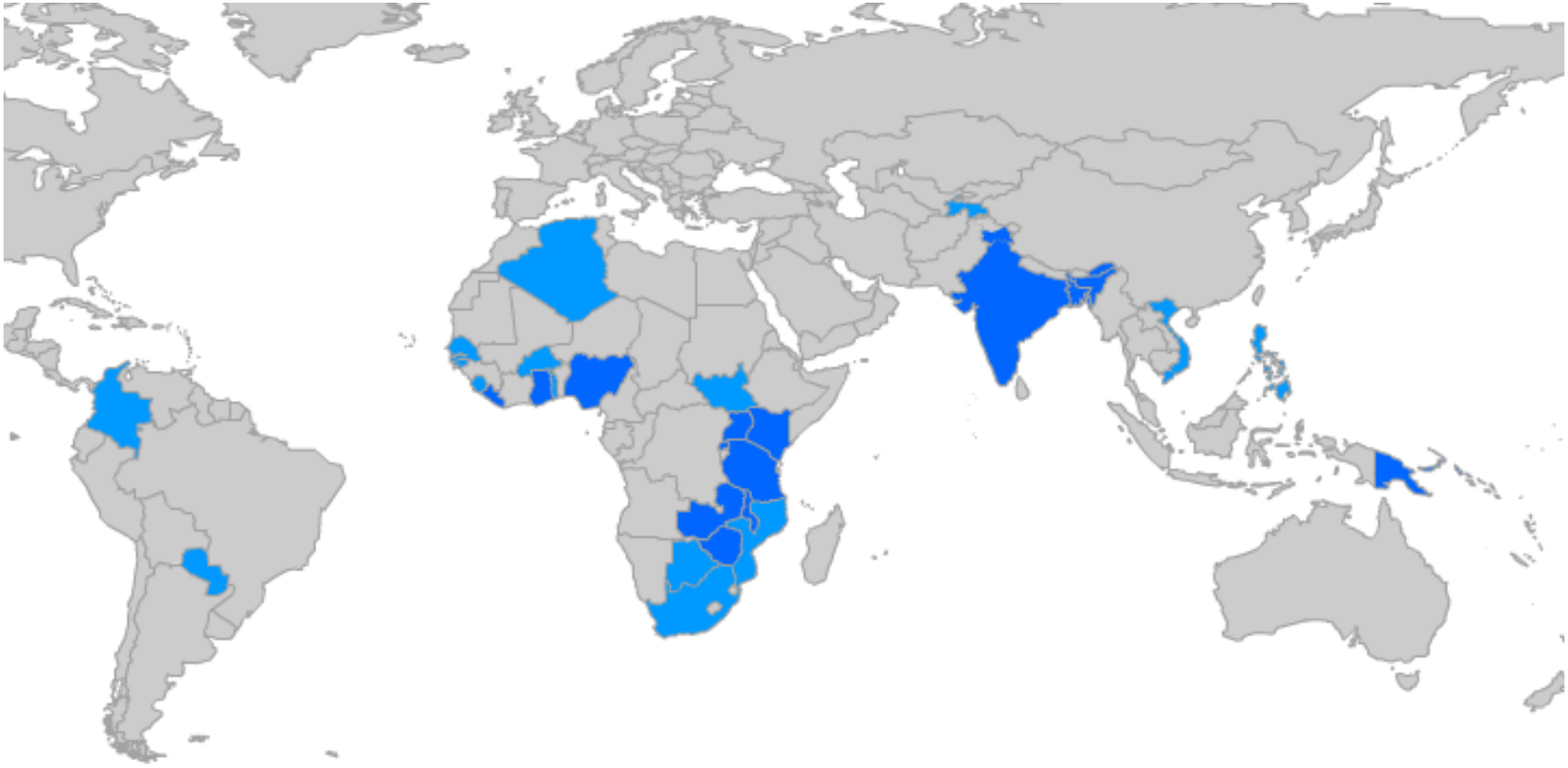
MobiMed-historien (2)

- Anestesisykepleier fulgte Halden-ambulansene.
  - Fra januar 1999: tillatelse til å gi trombolytisk behandling under transport (på legens ordre)
- Juni 2000: Mer enn 400 EKG hadde blitt sendt, total "call-to-needle-time" var redusert med gj.snittlig 50-60 minutt.
- Slutt på frivillig bruk: kardiolog på vakt blei pålagt å tolke innsendte EKG.

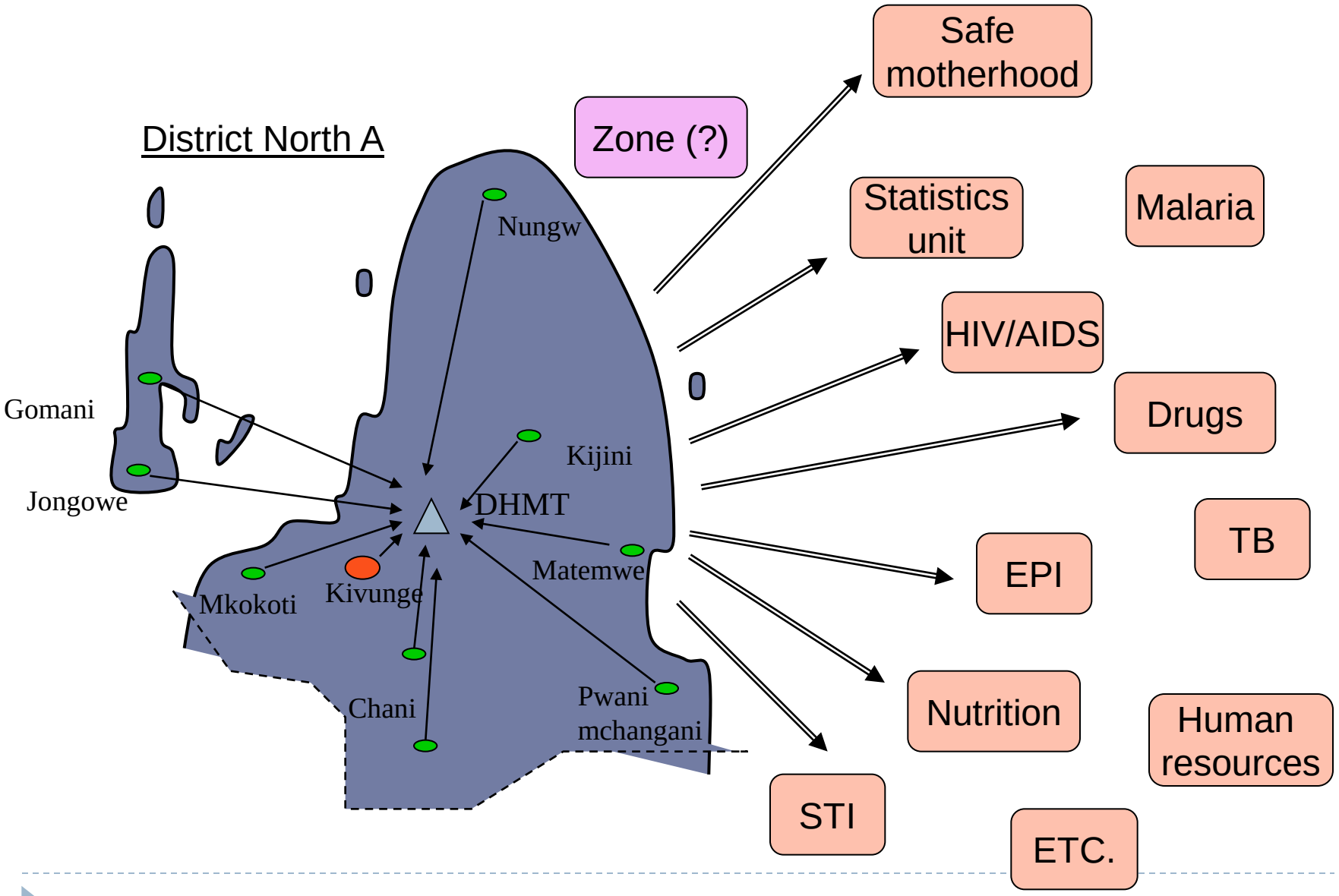
MobiMed-historien (3)

- 1999: Askim mister akuttmottaket
  - April 2000: MobiMed i Askim-ambulansar
  - Oktober 2000: Mer enn 200 EKG er sendt, sykepleier gir medikament
- Ambulanse-personellet når nivå 3 i sin utdanning (nasjonalt initiativ) og kan gi medikament
- 2001: MobiMed også i ambulansane i Sarpsborg, Moss og Fredrikstad

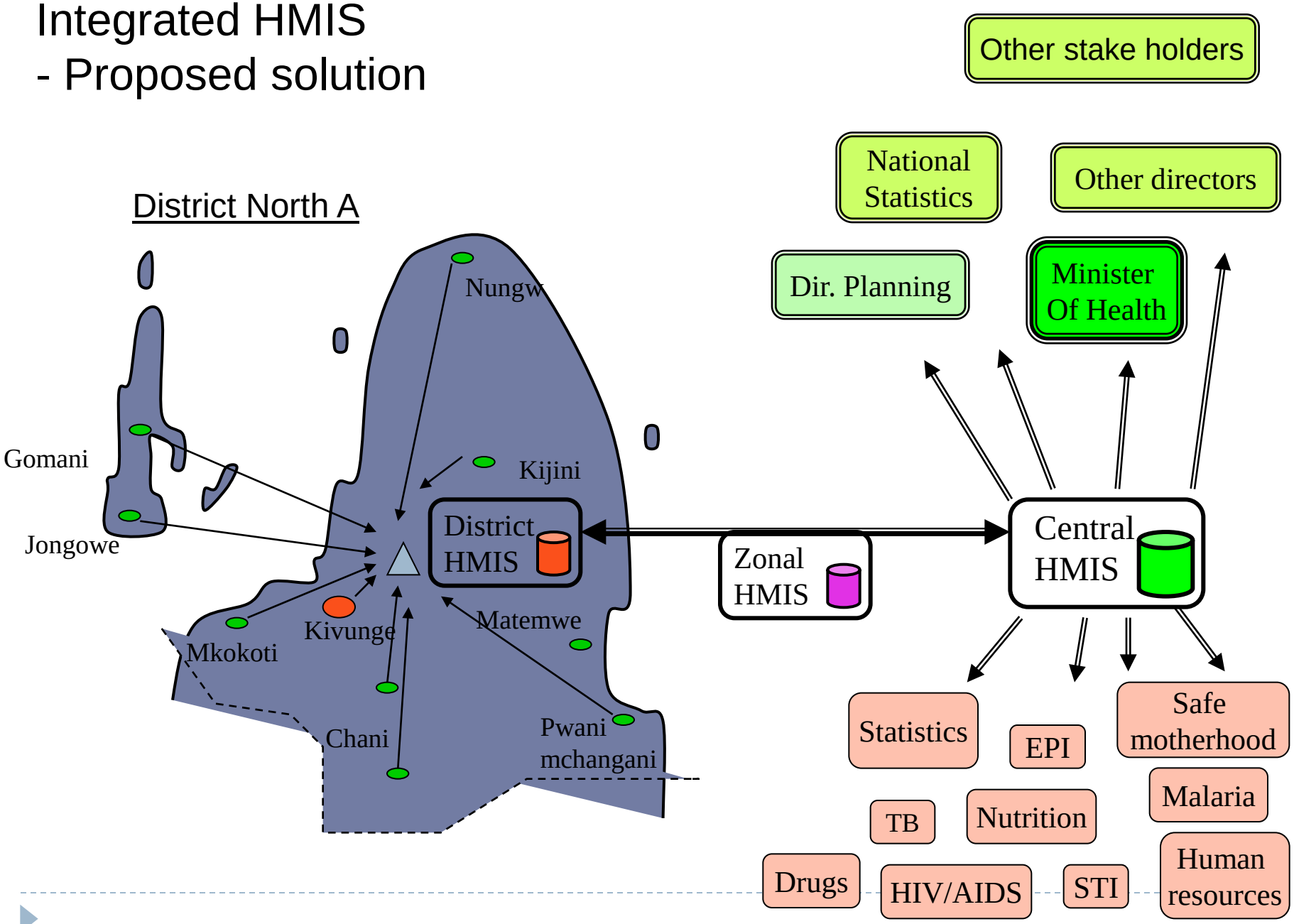
The Health Information Systems Programme (HISP)



AIM: Support local (district-level) decision making and utilization of information


District North A
Zone (?)
Safe motherhood
Statistics unit
Malaria
HIV/AIDS
Drugs
TB
EPI
Nutrition
Human resources
STI
ETC.
Nungw
Gomani
Jongowe
Kijini
DHMT
Matemwe
Mkokoti
Kivunge
Chani
Pwani mchangani

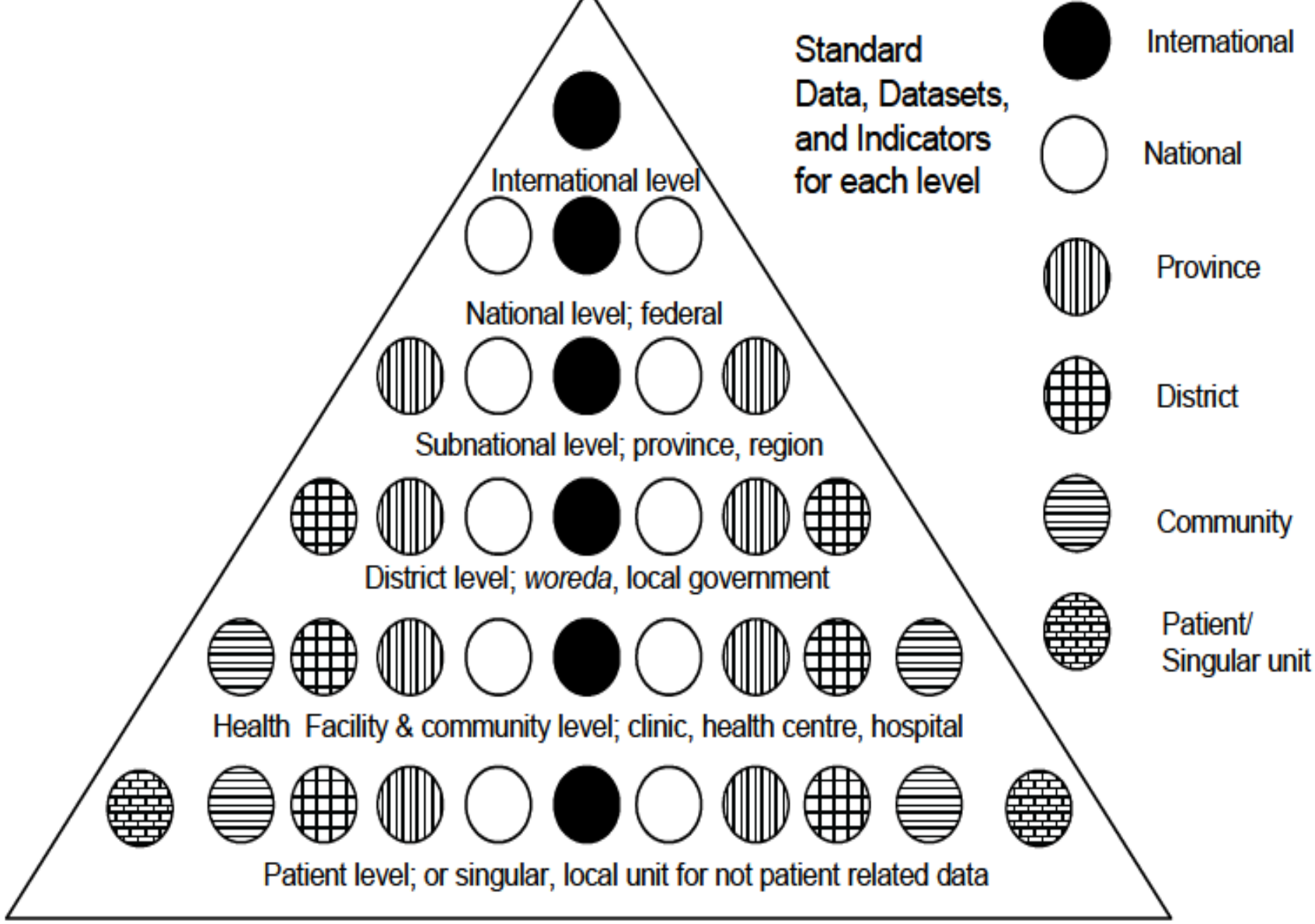
Integrated HMIS
- Proposed solution



Artikkel: Flexible standardization

- Standard for data-sett (”attractor”)
  - Eastern + Western Cape -> Sør-Afrika (2000)
- ”Gateways”: ulikheter mellom geografiske områder i Etiopia -> ulike løsninger

- ”Radical change through small steps”
  - ”Use flexibility” (deal with heterogeneity)
  - ”Change flexibility” (remain adaptive)

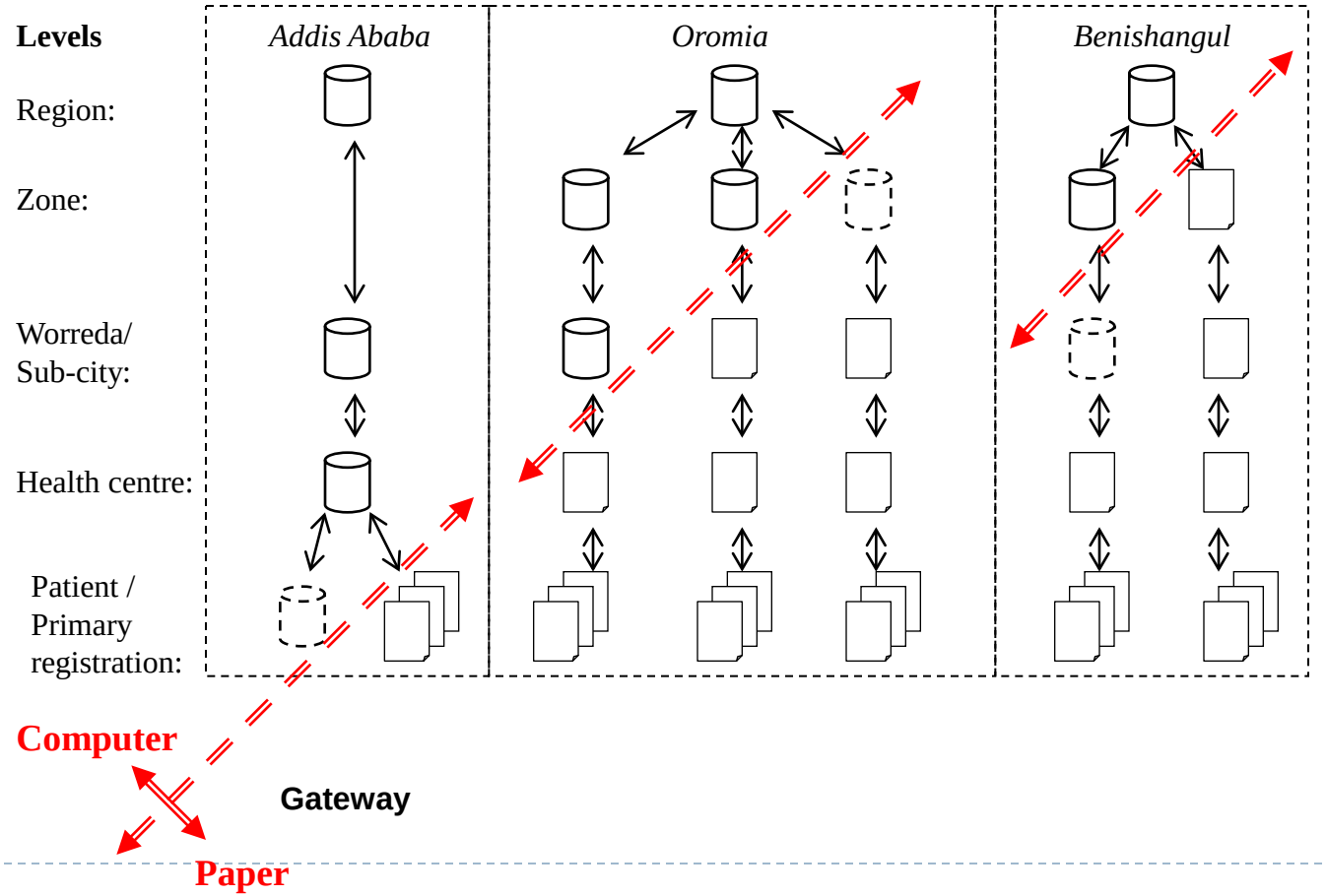


re 1. Hierarchy of Data Standards Used in South Africa with the Patient Level Added

Uneven development of Information infrastructure in developing countries requires flexibility in strategy:

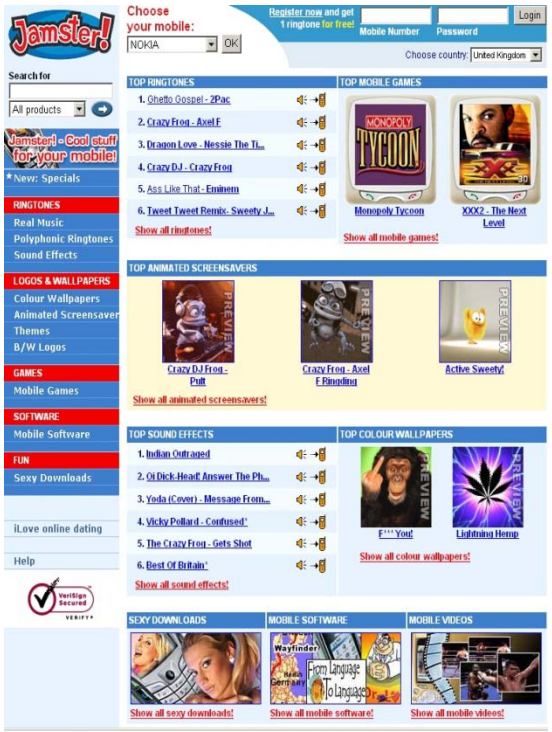
One size doesn't fit all - the technical solutions must be different

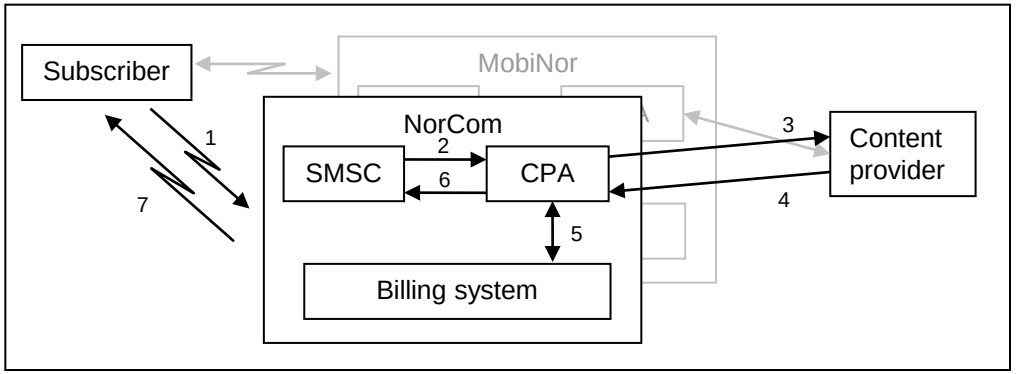
Standardisation must focus on data layer!!! The technical layers must be solved through ad-hoc gateways and flexible solutions (from web to donkeys!)



Nielsen og Aanestad: 'Control Devolution'

The Content Provider Access (CPA) Platform





The history of CPA (1)

- 1997: Early platforms for content services
  - Fixed prices (standard SMS price)
  - Exclusively for operator's own subscribers.
- Experiences:
  - Limited traffic, high production and marketing costs
  - Restrictions on content, editorial responsibility
  - Some 'ad-hoc' coordination (codes and syntax)
  - Content providers pushed for integration and premium charging

The history of CPA (2)

- Despite the interest from content providers and subscribers:
  - No resources were allocated
  - No formal management decision – CPA was controversial (VAS)
  - Off-hours development, personal dedication, utilizing slack resources
- Spring 2000: Launch of market wide Content Provider Access Platform (again pushed by content providers)
  - From a proprietary system under the operators' control to open access for content providers
  - Redefinition rather than major redevelopment

The history of CPA (3)

- *"Infrastructuralization"* – pushing the system "down" and "out".
- Costs turned out to be marginal, while traffic increased rapidly: turnover of approx 1 billion NOK (125 million €) in 2004.
- This move reconfigured the business sector information infrastructure:
- Operators
  - Avoided the responsibility for the content and escaped regulation
  - Can leave the development and provision of Value Added Services to others and focus on core areas
  - Maintain technical control (transportation and billing)
- Content providers
  - Have easy access to the whole market
  - Have responsibility for their own services
  - Can easily reconfigure and adapt the services
- Aggregators and application/software houses
  - Emerged 'between' the tele-operators and the content providers
  - Administrative work + building add-ons to the platforms

Infrastructuralization as design strategy

- Deliberate or accidental 'design'?
  - Happened outside the gaze of management, and 'below the consciousness level' of the organization, as radical innovations tend to do
- 'Accidentally' disclaiming control
  - Services not central enough to warrant attention and resources
  - Only possible with initially proprietary technical implementations but "walled garden" approach not long-term economically viable
  - Lack of strategic focus allowed a role for external actors
- 'Deliberately' disclaiming control
  - Editorial control (regulation, public perception, production costs)
  - Maintaining control: e.g. technical control resides with operators
  - Striking a certain balance between control and autonomy

Kontroll eller fleksibilitet?

- Planlegging og estimering på forhånd (kontroll)
  - versus
- å skape aksept og mobilisere ressurser mens man utvider (fleksibilitet)

- Diskusjon:
  - Når (for hva slags prosjekt) passer hvilken fremgangs-måte?