Granskning av Meddelandetjänst 1.0

Version: PA1

Datum: 2013-11-14

Innehåll:

TOC \o 2-3 \t "Heading, 4"

Inledning PAGEREF \_Toc \h 1

Granskningsresultat PAGEREF \_Toc1 \h 2

Licensform och licensheader i källkodsfiler PAGEREF \_Toc2 \h 2

Dokumentation på wiki’n PAGEREF \_Toc3 \h 2

Arkitekturbeskrivning (SAD) PAGEREF \_Toc4 \h 2

Instruktioner för utvecklare PAGEREF \_Toc5 \h 3

Instruktioner för lokal testning via jetty PAGEREF \_Toc6 \h 3

Driftinstruktioner (loggning och övervakning) PAGEREF \_Toc7 \h 3

Folder, maven-modul och paketstruktur PAGEREF \_Toc8 \h 4

Tester (manuella och automatiska) PAGEREF \_Toc9 \h 5

Automatiska tester PAGEREF \_Toc10 \h 5

Manuella tester PAGEREF \_Toc11 \h 6

Bygg och release process PAGEREF \_Toc12 \h 6

Överenstämmelse mellan källkod och implementationsvy i SAD PAGEREF \_Toc13 \h 6

Övergripande genomgång av källkoden PAGEREF \_Toc14 \h 6

# Inledning

Granskning utförd 20-13-11 baserad på:

1. Källkod Git på <https://code.google.com/p/inera-message/>, master-branch med commit id: 0eb5c29e8896b388f2b35d276bb1d4106a207d54 [0eb5c29]
2. Dokumentation på Wiki-sidan <https://skl-tp.atlassian.net/wiki/pages/viewpage.action?pageId=11436041>

Följande aspekter har granskats:

* Licensform och licenstest i källkodsfiler
* Del av dokumentation på wiki’n
  + Arkitekturbeskrivning (SAD)
  + Instruktioner för utvecklare
  + Instruktioner för lokal testning via jetty
  + Driftinstruktioner (loggning och övervakning)
* Folder, maven-modul och paketstruktur
* Tester (manuella och automatiska)
* Bygg och release process
* Överenstämmelse mellan källkod och implementationsvy i SAD
* Övergripande genomgång av källkoden

Följande aspekter har ***inte*** granskats (primärt pga tidsbrist):

* Detaljerade kodstudier
* Del av dokumentation på wiki’n
  + Övergripande texter och struktur på wiki’n för MT
  + Driftinstruktioner (installation och konfiguration), täcks upp av Hans.
  + Testinstruktioner och testresultat, täcks upp av Johanna.

# Granskningsresultat

## Licensform och licensheader i källkodsfiler

Korrekt licensform och genomgående korrekta licensheader i källkodsfiler.

**Anmärkning:** Licensheadrar anger felaktigt Sjukvardsradgivningen som IPR-ägare och inte Inera AB i följande filer:

- /mb-intsvc/src/test/resources/log4j.properties

- /mb-services/src/test/resources/log4j.properties

## Dokumentation på wiki’n

Följande delar har lästs igenom:

* Arkitekturbeskrivning (SAD)
* Instruktioner för utvecklare
* Instruktioner för lokal testning via jetty
* Driftinstruktioner (loggning och övervakning)

### Arkitekturbeskrivning (SAD)

**Not:** Se också stycket ”*2.6* Överenstämmelse mellan källkod och implementationsvy i SAD” för kommentarer om SAD’en.

**Anmärkning:** Kapitlet arkitekturella krav ange fortfarande ännu inte vilka felkoder som MT skall returnera vid ”kända fel” (tredje gången detta får en anmärkning i samband med granskningar…). För exempel se [EI SAD’s arkitekturella krav](https://skl-tp.atlassian.net/wiki/display/SKLTP/SKLTP+EI+SAD+-+Arkitekturella+krav).

**Anmärkning:** Svårt att förstå nuvarande innehåll i kapitlet "MT SAD - Designval"

**Anmärkning:** Dags att ta bort eller göra klart kapitlet "MT SAD - Följsamhet mot T-bokens styrande principer" (tredje gången detta kapitel får en formell anmärkning vid granskningar)

#### SAD - Implementationsvy

**Anmärkning:** Kapitlet i SAD’en som beskriver implementationsvy är minst sagt tunn och otillräcklig för att man skall få en förståelse för koden utan att behöva lägga massa tid på att detaljstudera källkoden. Se nedan för detalj-anmärkningar.

**Anmärkning:** Hittar ingen beskrivning i SAD’en impl vy av centrala mekanismer som transaktionshantering och felhantering samt mappning till standardiserade felkoder.

**Anmärkning:** Använding av korreleringsid nämns på något enstaka ställe i SAD’ens impl vy men ingen helhetsbild av dess användning och mekanism saknas.

(nämns i impl vy för ”spara meddelanden” men nämns inte någon annanstans?)

**Anmärkning:** I SAD’en beskrivning av ”hämta meddelande” nämns inte att de hämtade markeras i db som hämtade. För det är väl där det görs?  
Eftersom detta är av central betydelse för ”ta bort meddelande” så måste det väl beskrivas i ”hämta meddelande”?

**Anmärkning:** SA impl vy’s beskrivning av "ta bort meddelande" har en konstig formulering i regel 3 som gör att det inte går att förstå vad man menar.

### Instruktioner för utvecklare

Instruktioner lätta att förstå och följa.

**Anmärkning:** Dokumentation behöver uppdateras map att tester numer använder embedded resurser, dvs man behöver inte använda ActiveMq eller mysql i sin utvecklingsmiljö.

### Instruktioner för lokal testning via jetty

Läst igenom men ej testat i praktiken pga tidsbrist.

**Fråga:** Går det att köra detta mot embedded-resurser istället för externa resurser?

Nu när ni infört embedded-resurser i integrationstester så vore det ju väldigt tacksamt att kunna testa mot en exekverande instans i sin lokala maskin pss med embedded-resurser med t ex soap ui.

Så slipper man helt ifrån kravet på att ha aactiveMq och mysql installerade, konfigurerade och startade i sin lokala miljö.

### Driftinstruktioner (loggning och övervakning)

#### Loggning

Bra att loggning av vanliga händelser är dokumenterat.

**Anmärkning:** Det skulle vara väldigt värdefullt för support-personal om det också dokumenterades exempel på förväntade log utskrifter per händelse. Se [EI’s instruktion för loggning](https://skl-tp.atlassian.net/wiki/display/SKLTP/SKLTP+EI+-+Loggning%252C+Larm).

**Fråga:** Var finns rekommenderade lognivåer i QA och prod dokumenterade?

#### Övervakning

Bra att manuell övervakning finns dokumenterad.

**Anmärkning:** Det saknas beskrivning av hur automatiserad övervakning går till. T ex hur en övervakningsprogramvara kan detektera att det ligger meddelanden och väntar för lång tid.

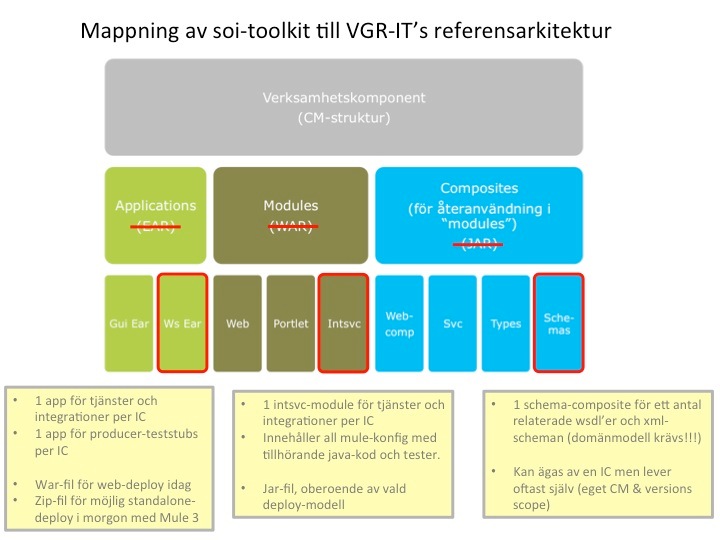
## Folder, maven-modul och paketstruktur

Följer standard folderstruktur för multi-module maven projekt.   
Korrekta java-paketnamn används.

**Anmärkning:** Applikationsmodulerna mb-services och mb-web (war-moduler) ligger fel, skall ligga under Applications, inte Modules. Se förtydligad bild av VGR’s ref-ark från 7 Nov 2011 i samband med att soi-toolkit anpssades till VGR’s ref-ark:

(filtyperna EAR, WAR och JAR är kopplade till ett Java EE deploy scenarie)

**Not:** Detta påverkar också bilderna i SAD’ens impl vy.



**Anmärkning:** Komponenten mb-webcomp är en typisk Module enligt VGR’s ref-ark, inte en Composite som den är placerad i nuläget.   
**Not:** Detta påverkar också bilderna i SAD’ens impl vy.

**Anmärkning:** Maven modulernas artifactId följer inte foldernamnens namn (vilket gör det svårt att mappa maven artifaktId till foldrar på filsystemet). Namnkonvention etablerad av EI skall följas!

T ex skall artifactId i "modules/ws/pom.xml" vara "mb-module-ws" och inte "mb-services" som det står nu.

**Anmärkning:** Maven-beroende till AMQ 5.8 används, NTjP har AMQ 5.6 i Test, QA och Prod. Backa till AMQ 5.6 och verifiera att ingen funktionalitet är bruten.

**Fråga:** Maven dependency management verkar inte användas genomgående utan mixas med versioner direkt på en del dependencies, varför?

Blir väldigt svårt att underhålla över tid.

**Fråga:** Vad är RivtaTkPublish/TD\_MESSAGEBOX\_1\_0\_0\_RC4?

(...varför RC4, är inte 1.0 releasad?)

**Anmärkning:** Källkod och jaxb-bindings-lösning i mb-schemas inte bra. Håll isär schema projekt och kod projekt! Varefter TK scheman publiceras på maven central repo så kommer denna kod att hänga i luften, då den inte kommer finnas i de TK-schema-komponenter som publiceras på maven central repo.

**Not:** Hade källkod och jaxb-bindings gällt lokala scheman som inte delas av andra så hade denna ansats så kalrt varigt helt ok.

**Anmärkning:** Inte ok att varken ta in soi-toolkit log-entry schema eller skapa egna klasser under soitoolkit's paketnamn!

Ta bort detta och använd befintlig soi-toolkit dependency för log-event-klasser!

**Fråga:** Varför används inte enbart Mavens Central Repository?  
Dvs, vad används <https://repository.jboss.org/nexus/content/groups/public> till?  
Lägg in en kommentar i pom.xml filen som förklarar!

## Tester (manuella och automatiska)

### Automatiska tester

Det finns såväl enhetstester om integrationstester som kan exekveras mha JUnit. Integrationstester kan exekveras mha inbäddade resurser, bra!  
  
Enligt projektet har testerna god coverage av kodbasen, dock saknas stöd för att automatiskt och repetitivt ta fram coverage rapporter (en brist i hela skltp’s kodbas för överigt).

**Anmärkning:** Följande felande tester måste rättas:

modules/ws/ReceiveMessagesIntegrationTest.receive\_ERR\_r2\_should\_return\_soap\_fault:92 expected:<2> but was:<1>

**Anmärkning:** Tester på mb-services i Eclipse ger fel:  
”SLF4J: Class path contains multiple SLF4J bindings.”

Rensa upp i slf4j config!

**Anmärkning:** Det loggas inget när man kör integrationstester i Eclipse (mb-services), så det är väldigt svårt att bilda sig en uppfattning hur långt en integrationstest kommit om den går fel. Log-konfiguration skall ändras så att definierade log-event på wiki’n loggas då integrationstester körs.

### Manuella tester

Se fråga i kapitel ”*2.2.3* Instruktioner för lokal testning via jetty”.

## Bygg och release process

Bygg och release process automatiserad mha Maven och CI-server finns på CloudBees.

**Fråga:** Var finns release processen beskriven?

**Fråga:** Hittar bara bygge triggat av commit i CI-server, inget nightly bygge. Varför?

**Anmärkning:** Samma fel stoppar bygget som när man startar de automatiska testerna manuellt, se ovan.

## Överenstämmelse mellan källkod och implementationsvy i SAD

**Anmärkning:** Hittar ingen beskrivning av designen av mb-webcomp, typ att Velocity används och hur. Designen måste beskrivas. Exempel på skärmdumpar av centrala användargränssnitt i SAD’ens implementationsvy vore väldigt behjälpligt för den oinsatte.

**Anmärkning:** Saknar beskrivning i SAD’en av centrala interface i MT, t ex se.skltp.mb.svc.services.

**Anmärkning:** Varför har man komplexiteten i metoden se.skltp.mb.intsvc.ReceiveMessagesImpl.extractTargetOrg()?  
VP sköter ju omvandling mellan RIV-TA 2.0 och 2.1.   
MT skall bara förhålla sig till RIV-TA 2.1 vad jag kan förstå.

**Anmärkning:** Borttag godkänns bara på redan lästa meddelanden.  
Hur garanterar man att det som markerat som lästa verkligen är lästa av konsumenten?  
SOAP Responset kanske aldrig når konsumenten?  
Därmed kan man väl felaktigt tillåta borttag av meddelanden som i praktiken aldrig lästs av konsumenten?

**Anmärkning:** Kan inte hitta dokumentation i SAD’ens impl vy angående riktlinjer för hur affärslogik placeras i koden.

T ex verkar det finnas en ”unread” kontroll i båda lagren intsvc och svc i respektive metod deleteMessages()?  
Bryter väl mot grundläggande DRY princip och därmed kostsamt och svårt att underhålla?

## Övergripande genomgång av källkoden

**Anmärkning:** Java doc i princip obefintligt (förutom standard källkodslicensfil) i impl-klasser. Delar av centrala interface i se.skltp.mb.svc.services är dock dokumenterade.

**Anmärkning**: Code guards vid loggning saknas, t ex

- logInfo(getLogger(), "Message " + msgId + " was deleted by " + callingSystem, msgId, msg);

I exemplet ovan kommer kostsam sträng objekt allokering ske för att bygga arguemnt två i metoden innan metoden logInfo() ens anropas. Detta kommer göras även om info-loggnign är avslagen. Ett väl etablerat mönster för att eliminera denna typ av resursslöseri är att anvädns en guard i koden typ:

**if (log.isInfoEnabled())** logInfo(…);

Kodbasen behöver gås igenom och rättas.

**Anmärkning:** message.properties i mb-webcomp är i UTF-8 men skall vara ISO-8859-1 (dvs standard för Java proeprtyfiler).

Se t ex <http://docs.oracle.com/javase/7/docs/api/java/util/Properties.html> och <http://docs.oracle.com/javase/7/docs/api/java/util/PropertyResourceBundle.html>.