Kuali Rice

> ⚠ Tämä selvitys on toteutettu kirjallisuusselvityksenä ilman arvioitavien ohjelmistojen koekäyttöä tai lähdekoodin katselmointia. Ohjelmistojen dokumentaatio on osittain ristiriitaista ja jättää monilta osin tulkinnan varaa. Näiden seikkojen vuoksi vahvistuksen saaminen kaikkiin selvityksessä esille tulleisiin asioihin ja niiden perusteella tehtyihin tulkintoihin vaatisi perusteellisempaa lähdekoodin katselmointia sekä ohjelmistojen testaamista käytännössä.

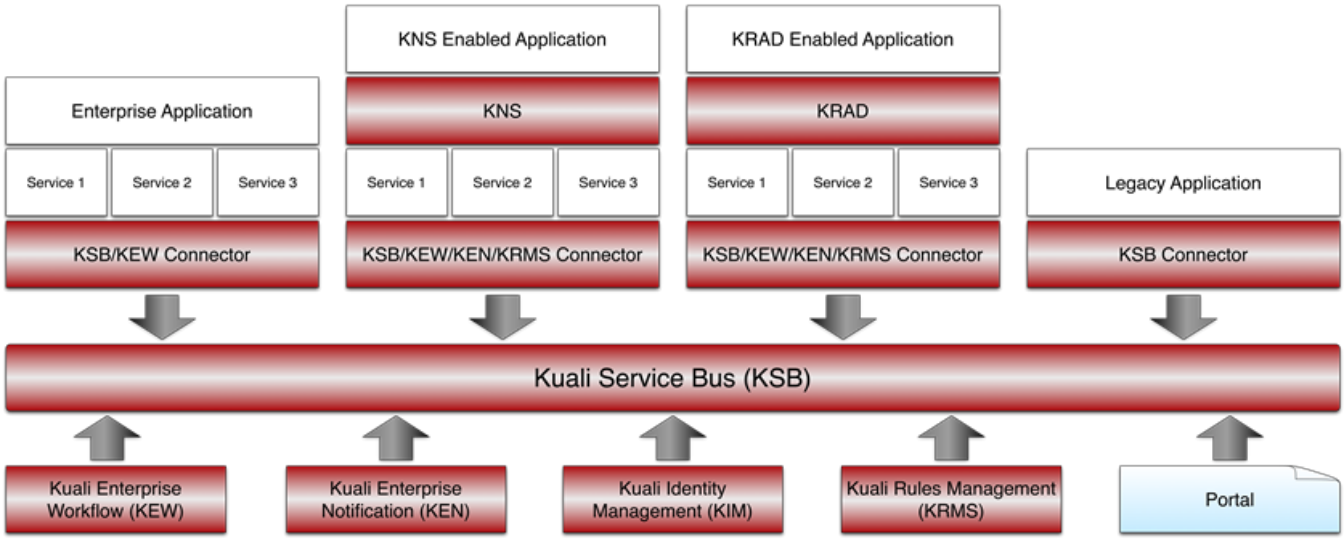
Kuali Rice (dokumentaatio) on useista eri moduuleista koostuva Java-ohjelmointikielellä toteutettu avoimen lähdekoodin middleware-ohjelmisto, joka on tarkoitettu sekä Kuali-ohjelmistoperheen ohjelmistojen että myös muiden ohjelmistojen käyttöön. Ricen toteutuksessa on pyritty palvelukeskeiseen arkkitehtuuriin (SOA) eli uudelleen käytettäviin tietojärjestelmäpalveluihin. Käytännössä Rice pitää sisällään sekä palvelin- että klient-puolen komponentteja. Palvelinpuolella Rice on toteutettu Java-ohjelmointikielellä Spring-ohjelmistokehystä hyödyntäen. Klient-puolella Rice puolestaan tarjoaa joukon erilaisia palveluita sisältävän Java-ohjelmointikielelle tarkoitetun KRAD-ohjelmistokehyksen asiakassovellusten kehitystä helpottamaan.

Arkkitehtuuri

Rice on rakenteeltaan modulaarinen ja jokainen moduuli tarjoaa joukon palveluita Ricea käyttäville sovelluksille. Moduulit on liitetty toisiin palveluväylän välityksellä, jonka kautta niiden palvelut ovat myös muiden järjestelmien käytettävissä. Ricen seitsemän päämoduulia ovat:

- Kuali Enterprise Notification (KEN)
- Kuali Enterprise Enterprise Workflow (KEW)
- Kuali Enterprise Identity Management (KIM)
- Kuali Enterprise Nervous System (KNS)
- Kuali Rapid Application Development (KRAD)
- Kuali Rules Management System (KRMS)
- Kuali Service Bus (KSB)

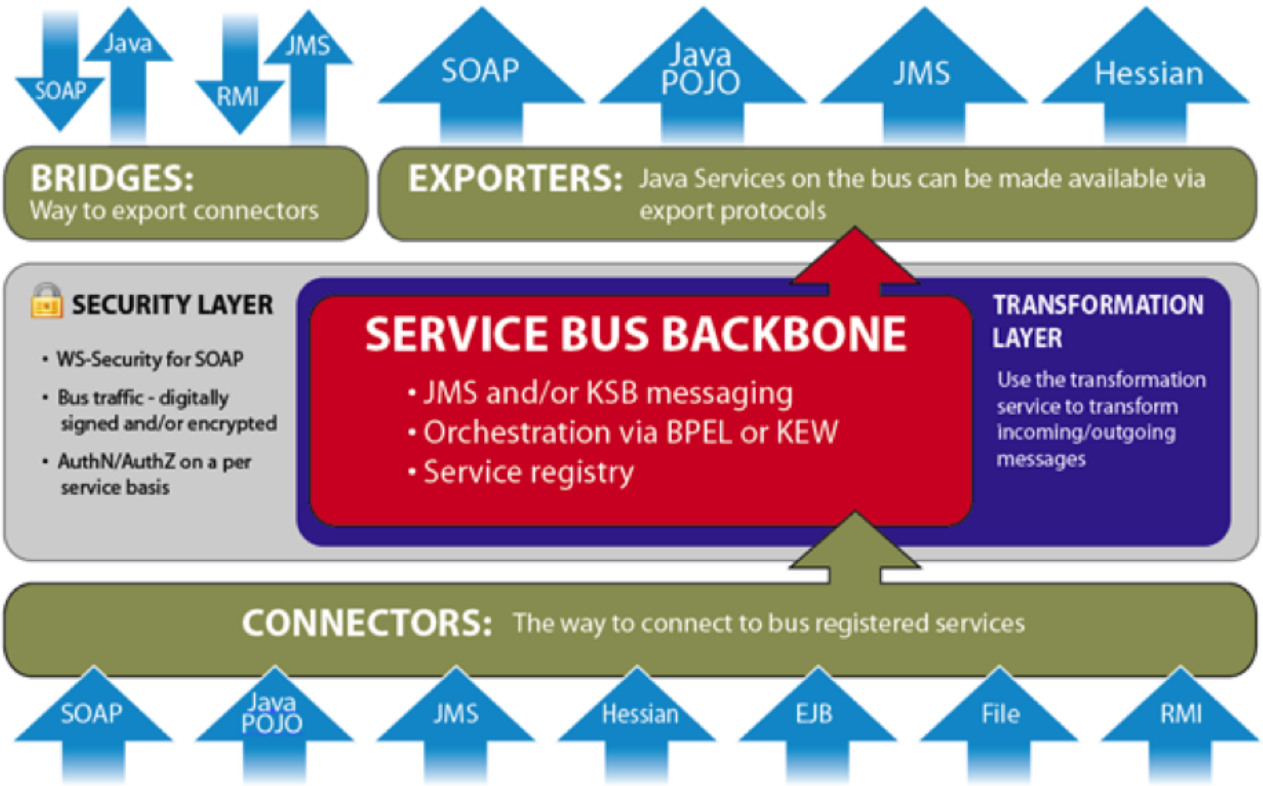
Kuvassa 1 on esitetty Ricen moduulien ja niitä käyttävien järjestelmien liittäminen toisiinsa palveluväylän välityksellä.



Kuva 1. Ricen moduulit ja niitä käyttävät järjestelmät.

Kuali Service Bus (KSB)

KSB on Java-ohjelmointikielellä toteutettu palveluväyläsovellus, joka mahdollistaa siihen liitettyjen palvelujen käytön sekä muista siihen kytketyistä järjestelmistä että ulkoisista järjestelmistä käsin. Järjestelmien liittäminen väylään on mahdollista sekä Spring-ohjelmistokehyksen XML-konfiguraatiotiedostojen kautta että ajonaikana ohjelmallisesti. KSB mahdollistaa siihen kytkettyjen järjestelmien kutsumisen ohjelmallisesti, JMS (Java Messaging Service) -viestien ja SOAP-viestien kautta. Palveluiden käyttö perustuu niille väylään liittämisen yhteydessä annettuihin nimiin. Tiedot väylään liitetyistä ja sen kautta käytettävissä olevista palveluista on tallennettu palvelurekisteriin. Rekisteriin tallennettuja tietoja hyödynnetään myös mm. kuormantasauksessa ja vikatilanteista toipumisessa. KSB:n rakenne on esitetty kuvassa 2.



Kuva 2. KSB:n rakenne.

KSB sisältää mm. seuraavat ominaisuudet:

- palvelurekisteri - palveluväylään kytkettyjen palvelujen kutsuminen tapahtuu palvelun nimen perusteella ilman, että palvelun fyysistä osoitetta tarvitsee tietää
- viestien välitys väylään kytkettyjen palvelujen välillä
- synkroniset ja asynkroniset viestit
- aihe-perustaiset viestit - väylään kytketyt palvelut voivat lähettää viestejä tiettyyn aiheeseen liittyen, jolloin viesti välitetään kaikille kyseisestä aiheesta kiinnostuneille palveluille. Viestin lähettäjä ei kohdista viestiä tietylle palvelulle, vaan viesti lähetetään tietylle aiheelle.
- vikasietoisuus - palvelukutsun epäonnistuessa palveluväylä yrittää kutsua muita saman nimisiä palveluita. Sekä synkroniset että asynkroniset viestit tuettuja.
- kuormantasaus - samaan palveluun kohdistuvat palvelupyynnöt jaetaan kyseisen palvelun eri instanssien kesken.
- viestijonot - väylään liitetyillä palveluilla on viestijonot, joihin palvelupyynnöt sijoitetaan. Mikä palvelu ei kutsuhetkellä pysty käsittelemään palvelupyyntöä, ei pyyntö katoa, vaan jää jonoon odottamaan käsittelyä.
- palvelupyyntöjen tallentaminen - kaikki palvelupyynnöt parametreineen on mahdollistaa tallentaa myöhempää tarkastelua varten
- tietoturva - tuki paveluväylän sisällä liikkuvien tietojen salaamiselle ja digitaalisten allekirjoitusten käytölle
- palvelun laatu - pavelupyyntöjen enimmäisvasteaika ja yritysten määrä säädettävissä.

Vikasietoisuuteen ja kuormantasaukseen liittyvät ominaisuudet perustuvat siihen, että väylään on kytketty useita saman palvelun eri palvelimilla ajettavia instansseja.

KSB:n ominaisuuksia voidaan tarkastella siihen liitettyjen erilaisia palveluita tarjoavien sovellusten (producer) ja tarjolla olevia palveluita käyttävien sovellusten (consumer) kannalta.

Palvelujen tarjoajat (producer)

KSB:n kautta on tällä hetkellä mahdollista julkaista vain Java-ohjelmointikielellä toteutettuja palveluita. Palveluiden on lisäksi rakennuttava KRAD-ohjelmistokehyksen päälle kyetäkseen hyödyntämään kaikkia Ricen ja KSB:n ominaisuuksia. RESTFul-rajapintojen julkaiseminen KSB:n kautta on myös mahdollista edellyttäen, että ne on toteutettu JAX-RS (Java API for RESTful Web Services) määrityksen mukaisesti. KSB mahdollistaa sen kautta julkaistujen palvelujen tarjoamisen ohjelmallisesti Java-ohjelmointikielellä sekä SOAP-viestien kautta.

Palvelujen käyttäjät (consumer)

KSB:n kautta julkaistujen palvelujen kutsuminen on mahdollista seuraavilla tekniikoilla:

- Java
  - Java POJO (Plain Old Java Object)
  - JMS (Java Messaging Service)

    - EJB (Enterprise JavaBeans)
    - RMI (Java Remote Method Invocation)
- SOAP (Simple Object Access Protocol)
- Hessian-protokolla

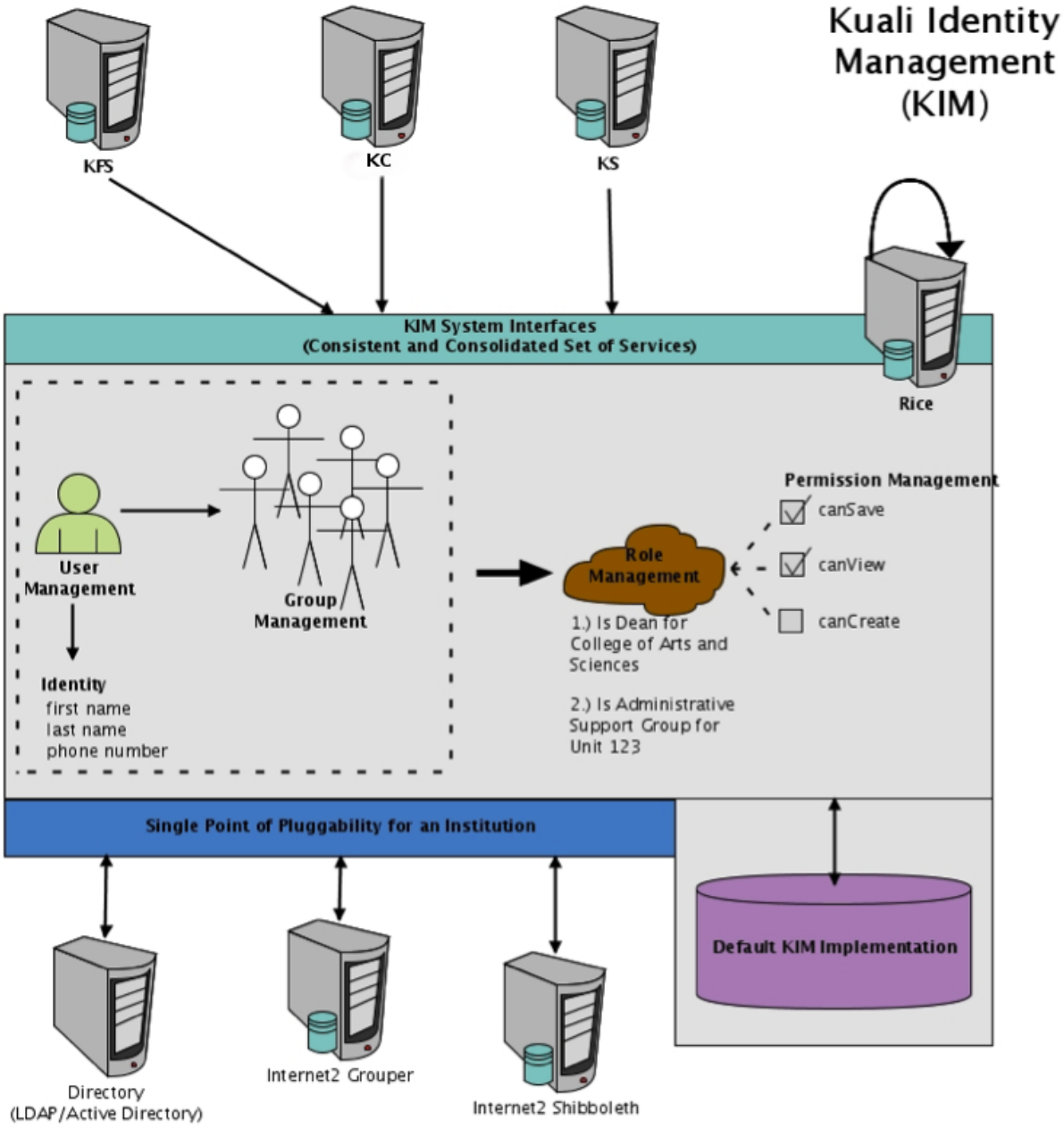
Vaikka KSB:n kautta julkaistut palvelut ovat kutsuttavissa SOAP-viestien kautta myös muista kuin Java-sovelluksista, niin kaikki KSB:n ja Ricen ominaisuudet ovat kuitenkin käytettävissä vain Javalla toteutetuissa KRAD-ohjelmistokehyksen päälle rakennetuissa sovelluksissa. Vain KRAD-ohjelmistokehykseen perustuvien sovellusten käytettävissä olevia ominaisuuksia ovat mm. asynkronisten viestien tuki, koosteisten palvelujen transaktioiden hallinta ja publish-/subscribe-viestit. Muut kuin Javalla toteutetut sovelluksen voivat kutsua KSB:n kautta julkaistuja SOAP-palveluita, mutta ne eivät pysty käyttämään KSB:n palvelurekisteriä palveluiden paikantamiseen. Tältä osin KSB:n ja Ricen dokumentaatio on kuitenkin ristiriitainen, sillä tietyissä osissa annetaan ymmärtää, että palveluiden paikantaminen ja rekisteröiminen KSB:hen on mahdollista erillisen SOAP-palvelun kautta.

Kuali Enterprise Enterprise Workflow (KEW)

KEW mahdollistaa elektronisten dokumenttien luontiin, reititykseen ja prosessointiin liittyvien työprosessien mallinnuksen ja toteuttamisen KEW:iin luotujen prosessikuvausten pohjalta. Kuvaus pitää sisällään työprosessin eri vaiheet sekä niihin liittyvät järjestelmät ja käyttäjät. KEW on vastuussa koko prosessin läpiviemisestä sekä kaikkien prosessiin liittyvien järjestelmien ja käyttäjien kanssa kommunikoimisesta kuvauksen edellyttämällä tavalla. Käytännön tasolla työprosessien hallinointi tapahtuu käsiteltävien dokumenttien sisältöön ja prosessikuvauksiin perustuvan reitityksen kautta. Reititys perustuu siihen, että KEW tunnistaa missä prosessin vaiheessa käsiteltävä dokumentti on, jonka jälkeen se reitittää dokumentin prosessikuvauksen määrittämälle seuraavalle vastaanottajalle. Loppuun saatettujen dokumenttien tiedot eri vaiheineen ja käsittelijöineen tallentuvat KEW:iin, jonka ansiosta jokaisen dokumentin täydellinen historia on nähtävissä järjestelmästä jälkeenpäin. Järjestelmän käyttäjille on mahdollista antaa vastuita, joita käytetään prosessikuvauksissa dokumenttien käsittelijöiden määrittelyssä.

Kuali Enterprise Identity Management (KIM)

KIM on käyttäjien ja käyttöoikeuksien hallintaan tarkoitettu sovellus, joka mahdollistaa käyttäjien, käyttäjäryhmien, roolien, käyttöoikeuksien, vastuiden ja edellä mainittujen välisten suhteiden hallinnan. KIM:in käyttö ja integrointi tapahtuu sen tarjoamien Java- ja WebService-rajapintojen kautta. KIM on toteutettu Java-ohjelmointikielellä ja sen toteutus perustuu Spring-ohjelmistokehykseen. KIM:in tarjoamat palvelut on mahdollista ottaa käyttöön kokonaisuudessaan sekä myös osittain, sillä sen yksittäiset palvelut on mahdollista korvata vastaavalla ulkoisella palvelulla. Käytännössä tämä tarkoittaa sitä, että esimerkiksi käyttäjätietojen ja käyttöoikeuksien tallentamiseen voidaan käytettää KIM:iä, mutta autentikointi toteutetaan jonkin ulkoisen järjestelmän kautta. KIM:in arkkitehtuuri on esitetty kuvassa 3.



Kuva 3. KIM:in arkkitehtuuri.

Kuali Enterprise Notification (KEN)

KEN mahdollistaa eri muotoisten viestien lähettämisen KSB:hen kytkettyjen järjestelmien ja niiden käyttäjien välillä. Oletusarvoisesti viestit näytetään KSB:hen kytkettyjen järjestelmien omissa käyttöliittymissä, mutta käyttäjät voivat halutessaan saada valitsemansa tyyppiset viestit myös esimerkiksi sähköpostiin tai mobiililaitteeseen. Vaihtoehtoisesti käyttäjille voidaan myös lähettää sähköpostiin tai mobiililaitteeseen tieto uuden viestin saapumisesta, mikäli itse viestiä ei haluta lähettää järjestelmän ulkopuolelle. Käytännössä viestit toimitetaan käyttäjien työlistalle, jota käytetään myös KEW:in työprosessihin liittyvien viestien ja tehtävien toimittamiseen. Tämän ansiosta käyttäjät näkevät kaikki heille lähetetyt viestit ja heidän vastuullaan olevat tehtävät yhdestä paikasta riippumatta viestin tai tehtävän tyypistä. KEN tarjoaa myös lähetettyjen viestien haku-toiminnon, joka mahdollistaa viestien hakemisen viestin tietojen perusteella. KEN:in tarjoama toiminnallisuus on käytettävissä sekä KSB:n että SOAP-rajapinnan kautta.

Kuali Enterprise Nervous System (KNS)

KNS on Rice-yhteensopivien web-sovellusten kehittämiseen tarkoitettu ohjelmistokehys, jonka kehitys on jo lopetettu. Jatkossa KNS:n korvaa KRAD.

Kuali Rapid Application Development (KRAD)

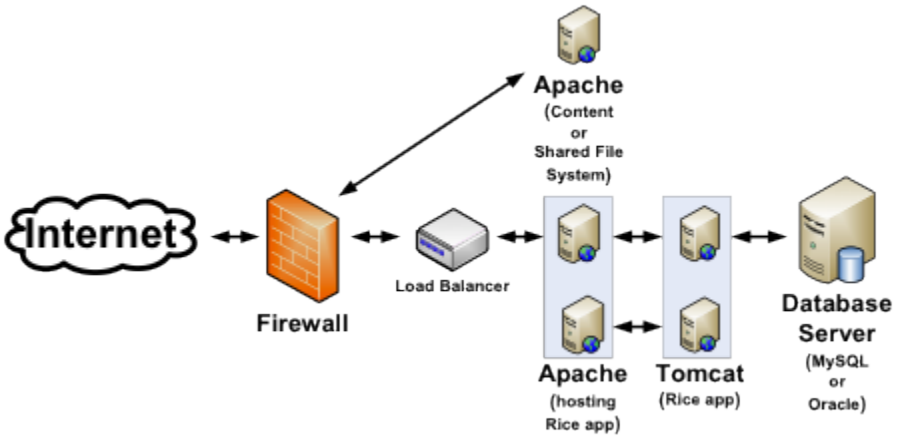
KRAD on Rice-yhteensopivien web-sovellusten kehittämiseen tarkoitettu ohjelmistokehys, joka helpottaa muiden Rice-komponenttien, kuten esimerkiksi KIM tai KEW käyttöä ja integrointia uusiin sovelluksiin. Ohjelmistokehyksen tarkoituksena on samalla nopeuttaa ja helpottaa Rice-yhteensopivian sovellusten kehitystä tarjoamalla valmiita ratkaisuja useilla eri sovelluskehityksen osa-alueilla. Ohjelmistokehyksen tavoitteena on yhtenäistää ja selkeyttää sitä hyödyntävien järjestelmien arkkitehtuuria ja tätä kautta myös helpottaa ylläpitoa ja jatkokehitystä. KRAD tarjoaa työkalut myös käyttöliittymien suunnitteluun ja toteutukseen, jonka ansiosta eri järjestelmien käyttöliittymät pystytään helpommin pitämään yhteneväisinä.

Kuali Rules Management System (KRMS)

KRMS on liiketoimintasääntömoottori, joka mahdollistaa järjestelmän toimintaa ohjaavien sääntöjen ja rajoitusten erottamisen järjestelmän ohjelmakoodista. Tällä tavoin liiketoimintasääntöjen muutokset on mahdollistaa toteuttaa ilman ohjelmkoodiin tehtäviä muutoksia, joka nopeuttaa ja yksinkertaistaa järjestelmien ylläpitoa. Käytännössä KRMS tarjoaa sääntöjen luomiseen ja ylläpitoon sekä tietenkin suorittamiseen liittyvän toiminnallisuuden. Useat KSB:hen kytketyt järjestelmät voivat hyödyntää samoja sääntöjä tai säännöt voivat olla myös järjestelmäkohtaisia.

Fyysinen arkkitehtuuri

Ricen fyysinen arkkitehtuuri eli kuinka järjestelmän osat käytännössä jakautuvat eri palvelimille on varsin joustava ja mahdollistaa useita erilaisia käytännön toteutuksia. Kehitysympäristössä Ricea ja kaikkia sen osia voidaan tarvittaessa ajaa yksittäisellä työasemalla, mutta tuotantoympäristössä Ricen eri osien hajauttaminen eri palvelimille on suositeltua. Kuvassa 4 on esitetty suositusten mukainen tuotantoympäristön arkkitehtuuri.



Kuva 4. Suositeltu tuotantoympäristön arkkitehtuuri.

Suositusten mukaisessa arkkitehtuurissa Rice on hajautettu useille eri palvelimille siten, että jokaista Rice-instanssia kohti on oma Apache-Tomcat-palvelinpari, jossa Apache toimii edustapalvelimena ja Tomcat Rice-instanssin alustana. Kaikki Rice-instanssit jakavat saman tietokannan, joka sijaitsee erillisellä tietokantapalvelimella. Käyttäjien pyynnöt eivät kuitenkaan mene suoraan Apache-palvelimelle, vaan käyttäjän ja Apachen välissä käytetään kuormantasaajaa varmistamaan pyyntöjen tasainen jakautuminen eri instanssien kesken. Edellä kuvattu kokoonpano on helposti laajennettavissa lisäämällä uusia Apache-Tomcat-pareja kuormantasaajan ja tietokantapalvelimen väliin.

Yhteenveto

Ricen toteutuksen lähtökohtana on palvelukeskeinen arkkitehtuuri, jonka keskeisenä ajatuksena on, että järjestelmä muodostuu toisiinsa löyhästi sidokissa olevista palveluista. Palvelujen tulisi olla mahdollisimman itsenäisiä sekä toisistaan riippumattomia, jolloin ne toimivat myös täysin itsenäisesti kokonaisuudesta erillään. Käytännössä yksittäisen palvelun siirtäminen järjestelmästä toiseen tai käyttäminen useassa eri SOA-järjestelmässä saman aikaisesti pitäisi siis olla mahdollista ilman palveluun tehtäviä muutoksia. Lisäksi SOA:n yhtenä keskeisenä periaatteena on myös palveluiden ohjelmisto- ja teknologiariippumattomuus. Palvelun toteutuksessa käytetyn tekniikan tai toteutusalustan ei pitäisi asettaa rajoitteita palvelua kutsuvalle järjestelmälle, vaan järjestelmien pitäisi pystyä kommunikoimaan keskenään käytetystä toteutustekniikasta riippumatta.

Ricen tapauksessa moduulit tarjoavat joukon erilaisia palveluita, jotka ovat yhteydessä toisiinsa Ricen oman KSB-palveluväylän kautta. Moduulien toteutuksessa on käytetty Kualin omaa KRAD-ohjelmistokehystä tai sen edeltäjää KNS-ohjelmistokehystä. KSB-palveluväylä on toteutettu siten, että sen kaikkien toimintojen hyödyntäminen on mahdollista vain Ricen omia ohjelmistokirjastoja käyttävistä järjestelmistä käsin. Muilla tekniikoilla toteutetut järjestelmät pystyvät hyödyntämään KSB:n kautta julkaistuja SOAP-palveluita, mutta eivät voi lainkaan julkaista omia palveluitaan muiden järjestelmien käyttöön. Käytännössä KSB:n käyttö integraatioalustana edellyttää, että ainakin kaikki sen kautta palveluita julkaisevat järjestelmät käyttävät hyväkseen Ricen ohjelmistokirjastoja.

Ricen omien moduulien toteutukset pohjautuvat Ricen ohjelmistokirjastoihin, jonka ansiosta niiden liittäminen KSB:hen on vaivatonta. Moduulien liittäminen jotakin muuta palveluväyläohjelmistoa käyttävään järjestelmään ei sen sijaan onnistu aivan yhtä helposti, koska moduulien rajapinnat on rakennettu Ricen ohjelmistokirjastojen tarjoamien rajapintojen varaan. Käytännössä yksittäisen moduulin liittäminen toiseen palveluväylään edellyttää, että moduuli on toisesta palveluväylästä huolimatta suoraan yhteydessä KSB:hen, jonka julkaiseman SOAP-rajapinnan kautta liittäminen toiseen palveluväylään tapahtuu. KSB:n poistaminen moduulin ja toisen palveluväylän välistä vaatisi rest- tai web sevices -rajapinnan rakentamista suoraan moduuliin. Moduulien ja Ricen välinen suhde on erittäin tiivis, sillä moduuleilla on ohjelmistokirjastojen kautta pääsy jopa suoraan Ricen tietokantaan, jonka lisäksi osa Ricen palveluista on sijoitettu keskitetyn Rice-palvelimen sijasta suoraan moduuleihin. Ratkaisun avulla on pyritty parempaan suorituskykyyn ja skaalautuvuuteen.

KEW, KRMS ja KEN ovat moderneja ratkaisuja, joiden tarkoituksena on erottaa ohjelmiston käyttöön liittyvät säännöt ja työnkulunhallinta ohjelmakoodista. Moduulien tarjoaman toiminnallisuuden avulla voidaan vaikuttaa ja säädellä järjestelmän toimintaa ilman ohjelmakoodiin tehtäviä muutoksia. KIM-moduuli puolestaan tarjoaa käyttäjiin ja käyttöoikeuksiin liittyvää toiminnallisuutta, joka pitäisi olla mahdollista ottaa käyttöön kokonaan tai vain tiettyjen toimintojen osalta. Kaikki edellä mainitut moduulit ovat tarjoamiensa toimintojen puolesta erittäin kiinnostavia ja voisivat periaatteessa olla hyödynnettävissä myös UKJ:n yhteydessä. Suurimmaksi kysymykseksi nousee moduulien vaatimien muutoksien määrä, jotta niitä voitaisiin tehokkaasti hyödyntää muuta palveluväylää kuin KSB:tä käyttävässä järjestelmässä.

KSB:n valinta UKJ:n palveluväyläksi ja integraatioalustaksi tarkoittaisi käytännössä UKJ:n moduulien teknisen toteutuksen sitomista Ricen ohjelmistokirjastojen käyttöön, koska KSB:n täysi toiminnallisuus on hyödynnettävissä vain niitä käyttävistä järjestelmissä käsin. Kokonaan alusta asti rakennettavien moduulien kohdalla tämä ei periaatteessa olisi ongelma, koska kirjastojen tarjoamat mahdollisuudet ja niiden asettamat rajoitukset voitasiin ottaa huomioon jo moduulien suunnittelussa. UKJ:n moduulien rakentaminen kirjastojen varaan vaikeuttaisi kuitenkin moduulien uudelleenkäyttöä muissa järjestelmissä, koska moduulien rajapinnat perustuisivat Riceen sidottuin ratkaisuihin, jotka eivät sellaisenaan ole yhteensopivia muiden järjestelmien kanssa. Muiden järjestelmien liittäminen KSB:n ympärille rakennettuun järjestelmäkokonaisuuteen vaikeutuisi myös huomattavasti, koska muut järjestelmät eivät olisi täysin yhteensopivia KSB:n kanssa ilman erilaisten järjestelmäkohtaisten välikappaleiden toteutusta. Muiden Rice-moduulien ja Kuali-ohjelmistojen liittäminen KSB:n ympärille rakennettuun järjestelmäkokonaisuuteen onnistuisi sen sijaan joustavasti yhteisen toteutustekniikan ansiosta.

Tulevaisuutta ajatellen on myös huomioitava, että Ricen ohjemistokirjastoja käytetään vain ja ainoastaan Kuali-järjestelmissä, joten sen käyttäjä- ja kehittäjäyhteisö on varsin rajattu moniin muihin avoimen lähdekoodin ohjelmistokehyksiin verrattuna, joita kehitetään ja käytetään eri puolilla maailmaa monissa eri tyyppisissä ja kokoisissa ohjelmistoissa. Ricen kapean käyttäjäryhmän vuoksi sen tuntevia kehittäjiä on myös erittäin rajoitetusti saatavilla, joka osaltaan hankaloittaa sitä käyttävien järjestelmien kehittämistä ja ylläpitoa.

Yleisesti ottaen UKJ:n moduulien teknisen toteutuksen sitominen tiettyyn toteutustekniikkaan tai ohjelmistokehykseen ei ole järkevää, vaan moduulit olisi pyrittävä toteuttamaan ohjelmisto- ja teknologiariippumaattomalla tavalla. Moduulien tarjoamien toimintojen tulee olla käytettävissä niiden tarjoamien avointen rajapintojen kautta riippumatta palvelun toteuttavan moduulin tai sitä kutsuvan järjestelmän toteutustekniikasta. Tällä tavoin moduulien toteutustekniikka voi vaihdella moduuleittain, jonka lisäksi myös yksittäisen moduulin korvaaminen toisella eri tekniikkaa käyttävällä toteutuksella on mahdollista, kunhan moduulin tarjoama rajapinta vain pysyy muuttumattomana.

Vaikka Rice pyrkiikin palvelukeskeisyyteen, niin käytännössä sen toteutus kuitenkin rikkoo monia palvelukeskeisyyden perusperiaatteita. Tästä ei kuitenkaan aiheudu ongelmia, kunhan järjestelmäkokonaisuus on rakennettu KSB:n ympärille ja siihen liitetyt järjestelmät käyttävät Ricen ohjemistokirjastoja. Kuali-järjestelmistä koostuva järjestelmäkokonaisuus toimii siis ongelmitta ja uusien Kuali-järjestelmien liittäminen kokonaisuuteen pitäis olla kohtuullisen vaivatonta. Ongelmalliseksi tilanne muuttuu silloin, kun KSB:n ympärille rakennettuun järjestelmään halutaan integroida muita kuin Kuali-järjestelmiä. Vastaavasti myös yksittäisten Kuali-järjestelmien integroiminen muihin järjestelmäkokonaisuuksiin ei ole ongelmatonta. Tämän perusteella KSB ei ole paras mahdollinen integraatioalusta, mikäli kokonaisuuden on tarkoitus pitää sisällään myös muita kuin Kuali-järjestelmiä. Ricen käyttö ohjelmistokehityksessä ei myöskään ole järkevää, jos kehitettävää ohjelmistoa halutaan käyttää yhdessä muiden kuin Kuali-järjestelmien kanssa. UKJ:n laajuuden ja tavoitteet huomioonottaen KSB:n käyttö palveluväyläratkaisuna ei siis vaikuta suositeltavalta. Tämä ei kuitenkaan sulje pois muiden Rice-moduulien ja Kuali OLE:n tai sen osien hyödyntämistä kokonaisuuden rakentamisessa. Hyödynnettävien osien UKJ:n tarpeisiin sovittamiseksi tarvittava työmäärä on kuitenkin arvioitava erikseen jokaisen käytettävän osan tasolla.