Analyse Activiti multi-tenancy

<http://www.jorambarrez.be/blog/2015/10/06/multi-tenancy-separate-database-schemas-in-activiti/>

* Activiti is sinds 5.15 multi-tenant compatibel. Verschillende instanties van de engine kunnen op 1 database met verschillende schema’s.
* Er wordt gebruik gemaakt van een tenantID.

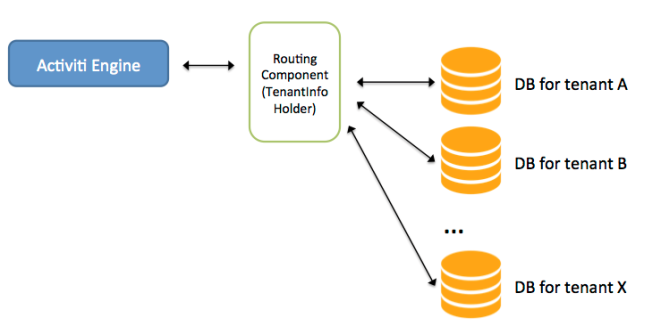
# Screenshot 2015-10-06 13.18.36Multi-Engine Multi-Tenancy

|  |  |
| --- | --- |
| Voordelen | Nadelen |
| Kunnen op verschillende fysieke servers draaien | Meer complexe set-up |
| Alle data zijn toegewijd aan slechts 1 engine | Verschillende configuratiefiles per tenant |
|  | Elke instantie neemt geheugen op, niet noodzakelijk efficiënt. |
|  | Routeringscomponent, om verkeer naar juiste tenant te sturen. Dus ook een soort tenantID |

Multi-Schema Multi-Tenancy

* 1 engine instantie ipv verschillende, verschillende databases
* tenants identificeren met een unieke tenantID
* Datasource implementatie per tenant, met eigen connection pool voor efficiënte resourcemanagement
* Code in “TenantAwareDataSource”, implementatie van javax.sql.DataSource die correcte datasource koppelt aan een tenantID

|  |  |
| --- | --- |
| Voordelen | Nadelen |
| Onderhoud, configuratie en API’s zijn net hetzelfde als multi-engine | Onderhoud en configuratie van verschillende databases |
| Data is gescheiden van andere tenants |  |
| Geen complexe engine management meer |  |
| Minder geheugen nodig |  |

* Routering gebeurt door dat er data van een instantie “TenantInfoHolder”

****

****