

AMW Features

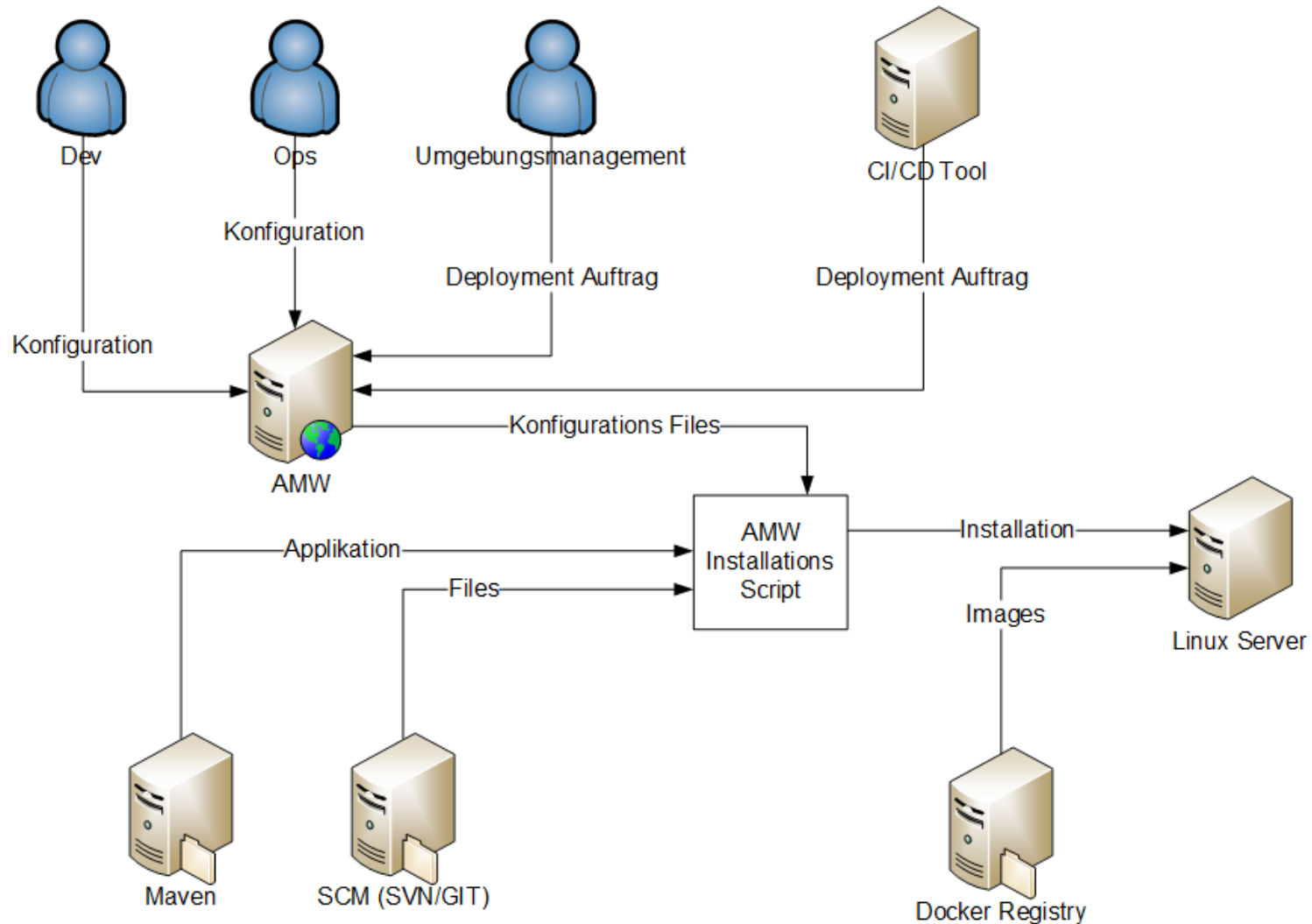
Was ist AMW? Welche Vorteile bietet AMW?

- **Automation Middleware (AMW)** umfasst die Konfigurationsverwaltung und das Deployment von Applikationen. Die Konfiguration wird unabhängig vom Applikationsserver erfasst, dadurch kann die gleiche Konfiguration für bestehende und zukünftige Applikationsserver verwendet werden.
- Vorteile von AMW
 - Konfiguration ist Produktunabhängig
 - Redundanzen von Konfiguration vermeiden
 - Versionierung/Auditing
 - Konfiguration und Applikation wird als ein Paket installiert
 - Konfiguration ist für andere Teams einsehbar, Rollenkonzept
 - Einfach erweiterbar: neue Schnittstellen oder Features

Die Kernfunktionen

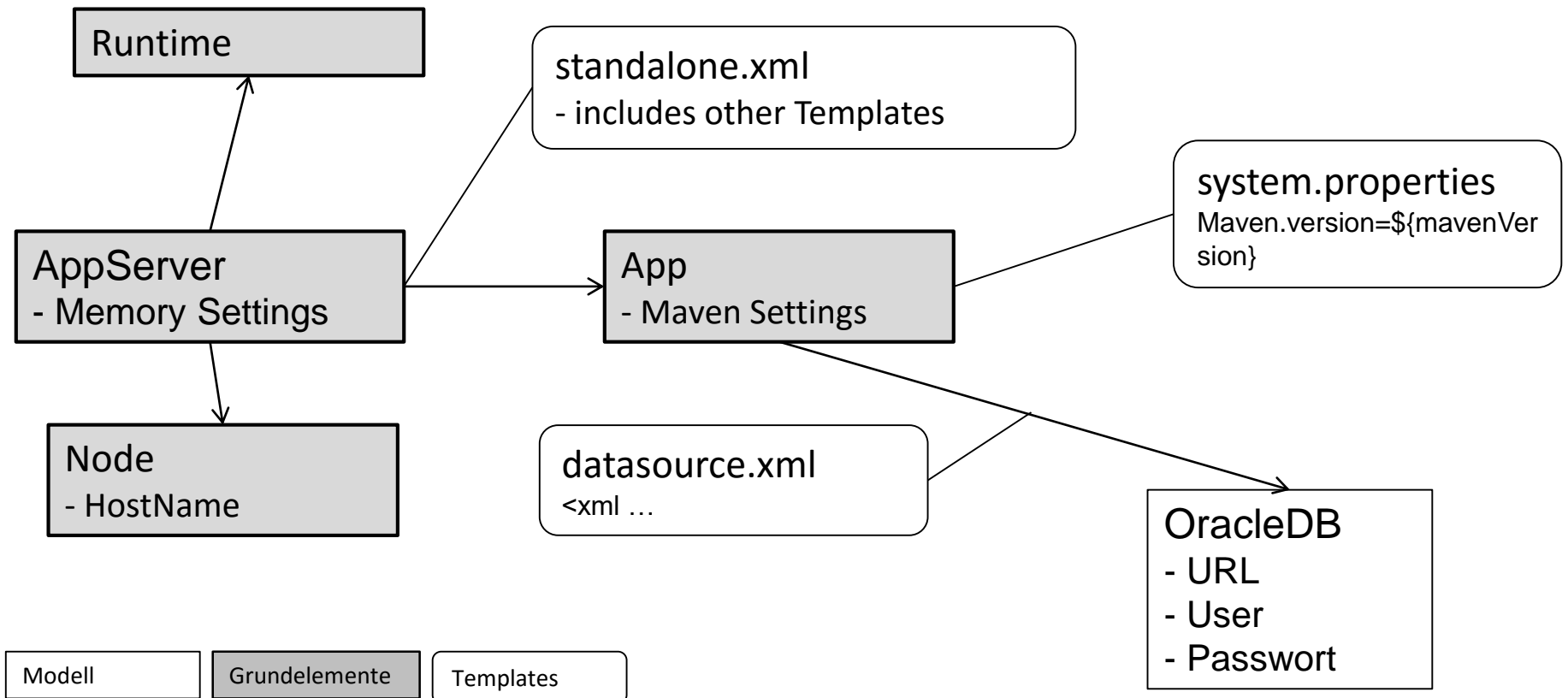
- Automatisierung & Standardisierung
- Verwaltung der:
 - Applikationskonfigurationen
 - Applikationsserverkonfiguration
 - Umsysteme & Ressourcen
- Deployment/Softwareverteilung der:
 - Applikationen
 - Applikationskonfiguration
 - Applikationsserverkonfiguration
- Versionierung & Nachvollziehbarkeit (Revision & Auditing)
- Validierung der Konfiguration (Qualitätssicherung)
- Inventar der Server
- Shakedown Tests

Deployment Prozess mit AMW



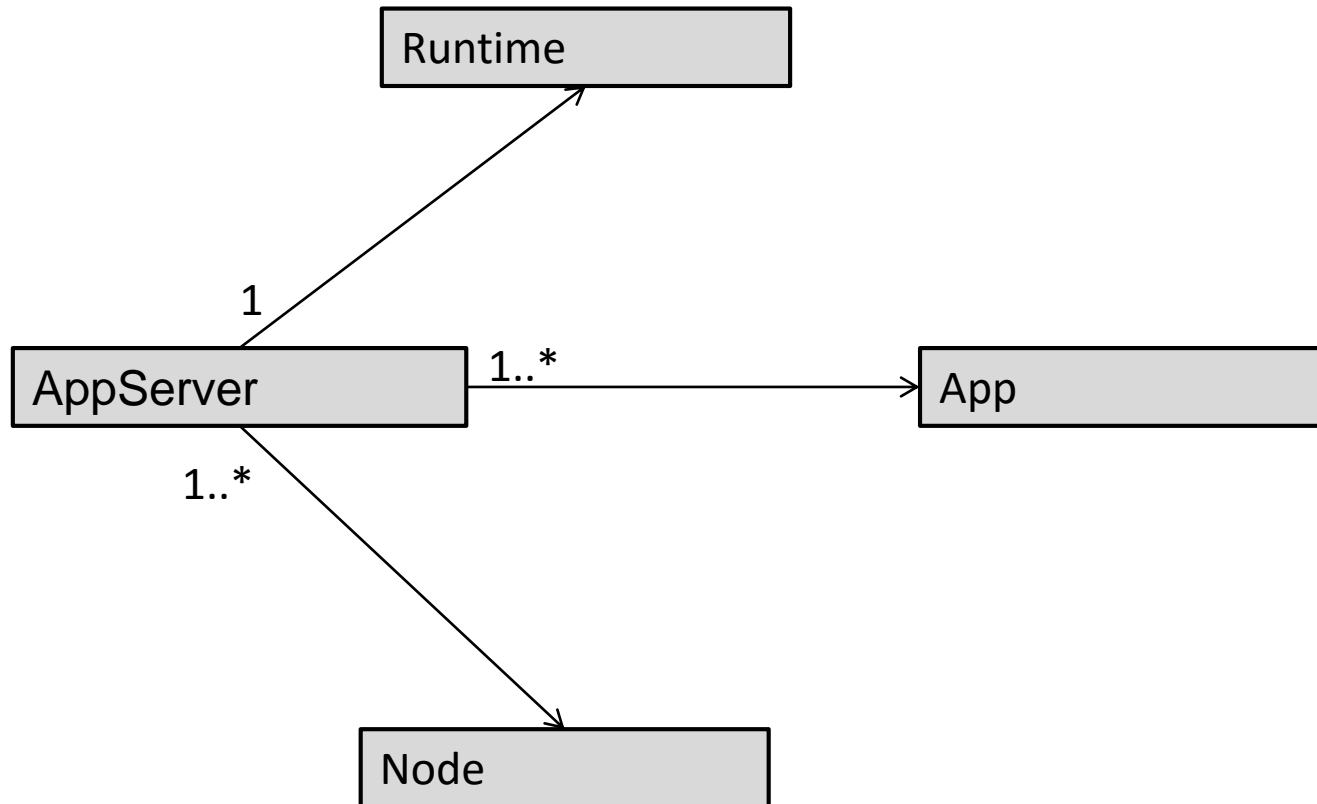
Modell

- Frei definierbares Objekt Modell: Ressourcen
- Modelliert eine Applikation mit seinen Abhängigkeiten
- Templates wandeln Properties via Freemarker Template Engine in Konfiguration um



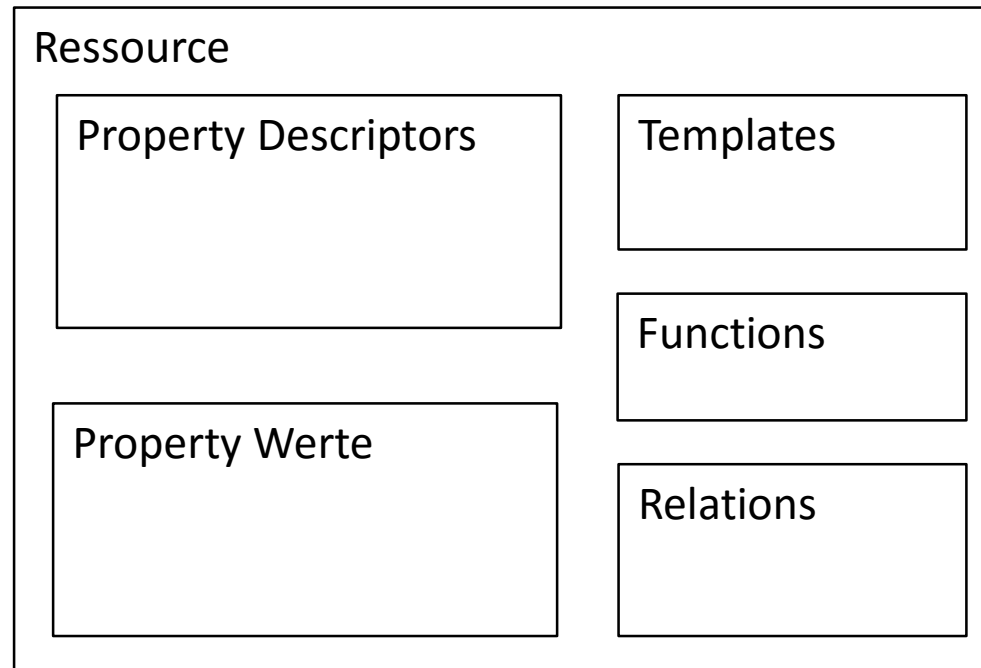
Modell: Default Ressourcen

- Default Ressourcen müssen immer vorhanden sein



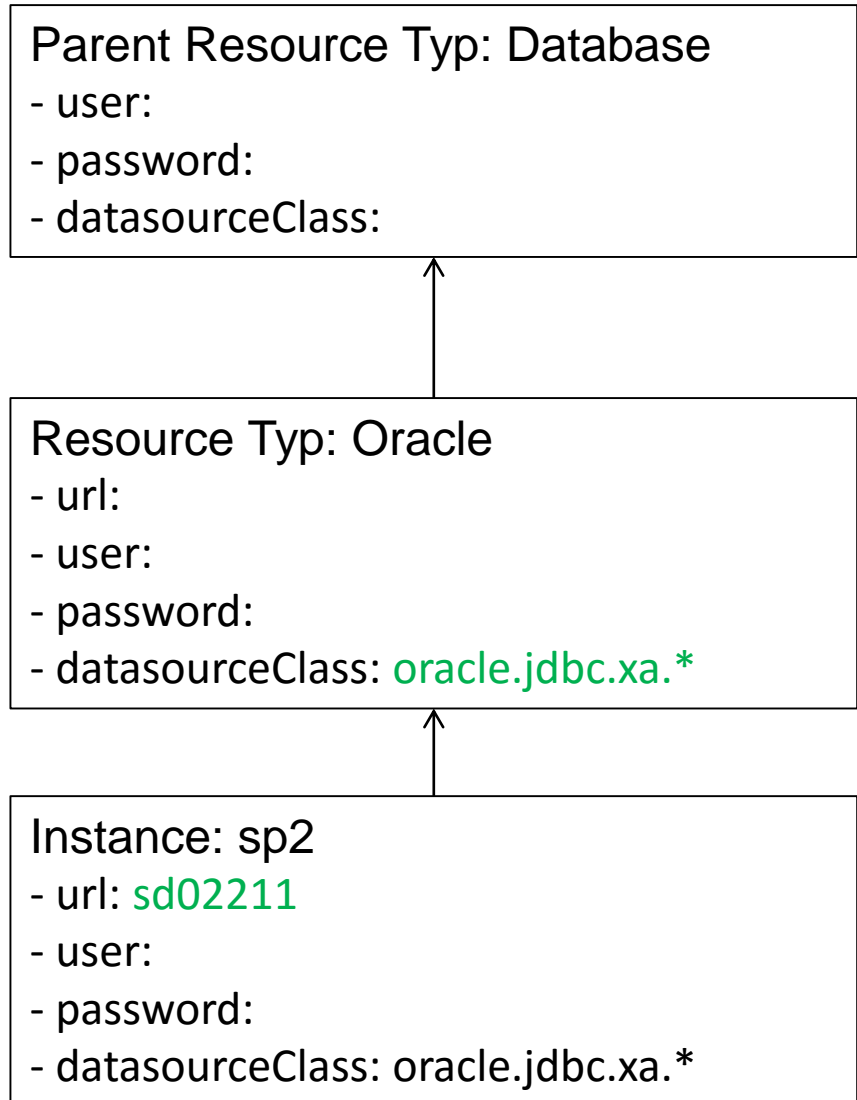
Modell: Ressource

- Elemente pro Ressource:
 - Properties Descriptors: Welche Properties gibt es?
 - Property Werte: Inhalt der Properties pro Umgebung
 - Templates: wandelt Properties in Konfiguration um
 - Funktionen: Zusammenstellung von Properties
 - Relations: Verknüpfung zu anderen Ressourcen für reuse



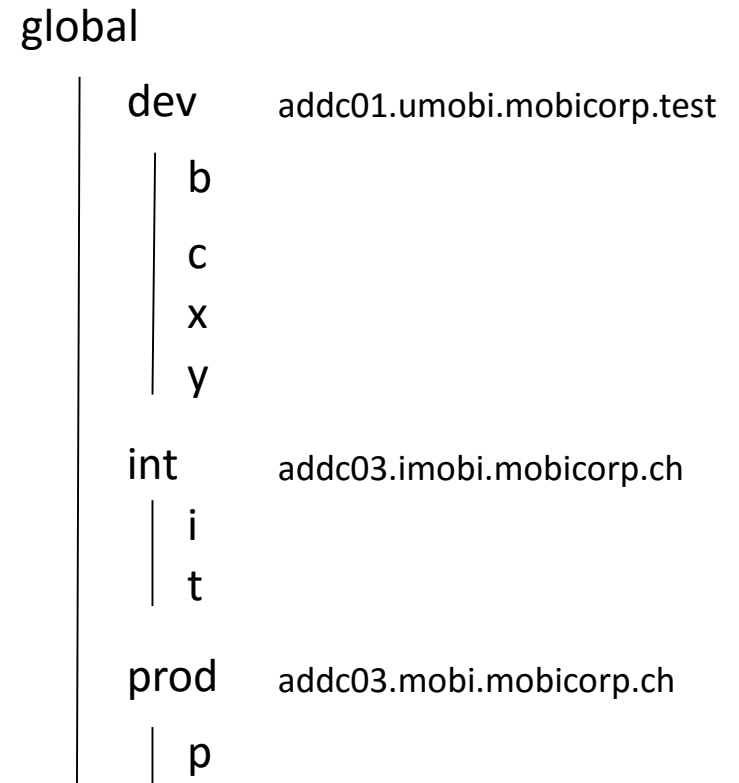
Modell: Vererbung & Properties

- Jede Ressource hat Typ
- Maximal ein weiterer Ober-Typ
- Letzte Ebene sind Instanzen
- Vererbt wird:
 - Property Descriptor
 - Property Werte
 - Templates
 - Funktionen
- Grün: auf dieser Ebene definiert
- Schwarz: auf höherer Ebene definiert



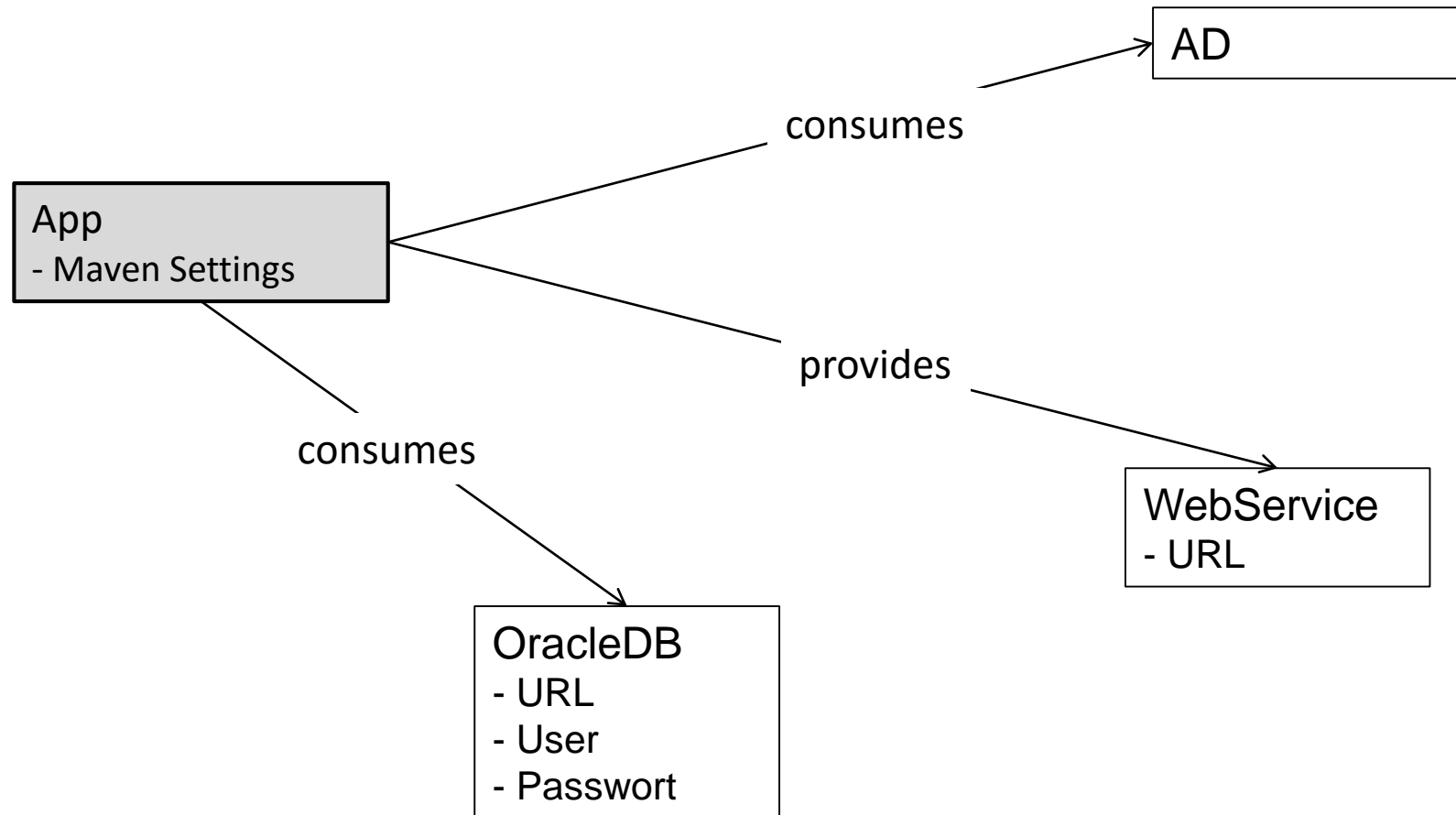
Modell: Umgebungs-Hierarchie

- Property Werten werden von Global nach Domain (dev, int, prod) nach Umgebung vererbt
- Ermöglicht Defaults
- Hilft Redundanzen vermeiden.
- Auf dem GUI:
 - **Grün**: auf dieser Ebene definiert
 - **Schwarz**: auf höherer Ebene definiert
 - **Rot**: Validierungsfehler
 - Im Tooltip (i) steht zusätzlich wo Property überschrieben wird.



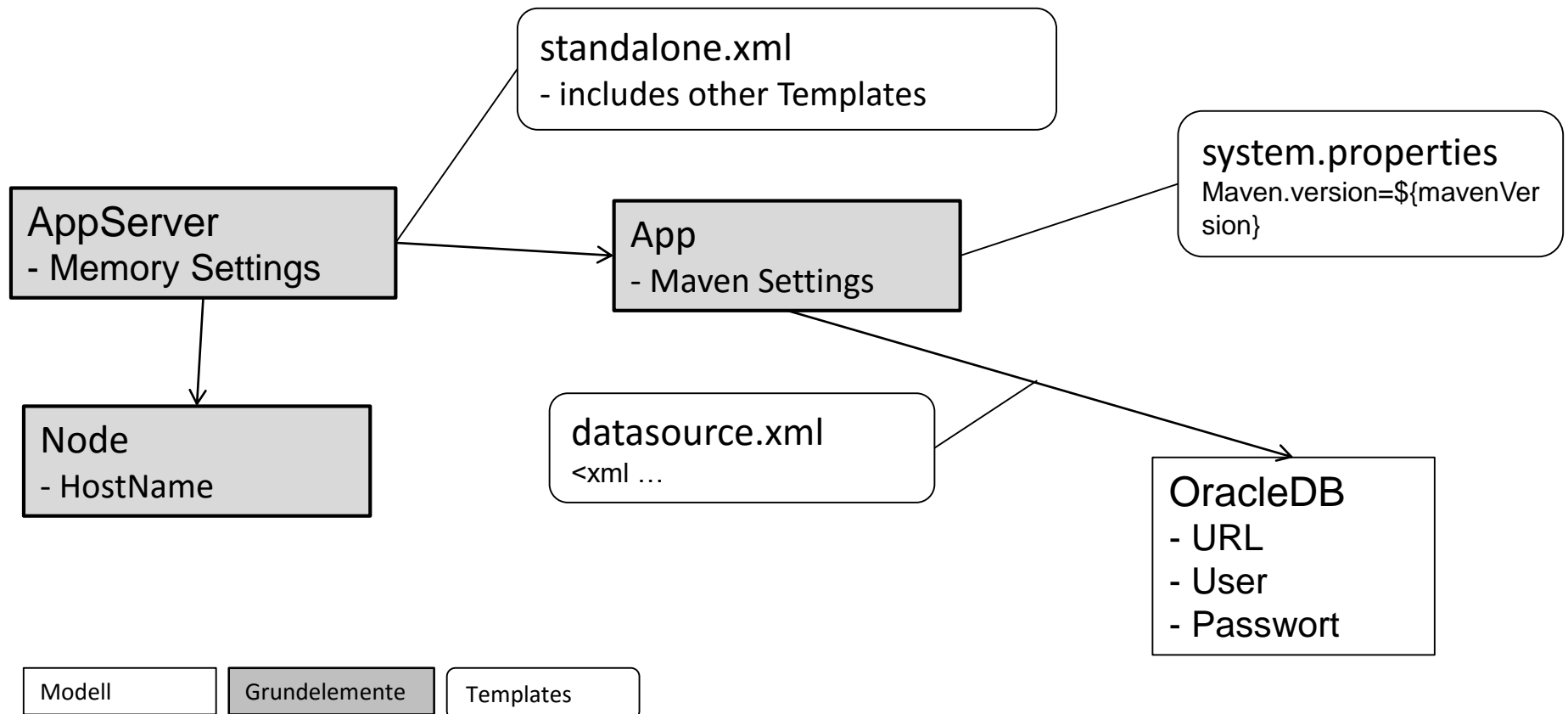
Modell: Relations

- Consumed: Ressource wird konsumiert
- Provided: Ressource wird angeboten
- Aus Templates kann auf die Related-Ressourcen Zuegriffen werden
- Properties können in der Relation überschrieben werden



Modell: Templates

- Templates dienen dazu, Konfigurationsfiles zu schreiben.
- Verwendet Freemarker: <http://freemarker.org/docs/>
- Template Position: Typen, Relations, Instanzen
- Template hängen an Runtimes

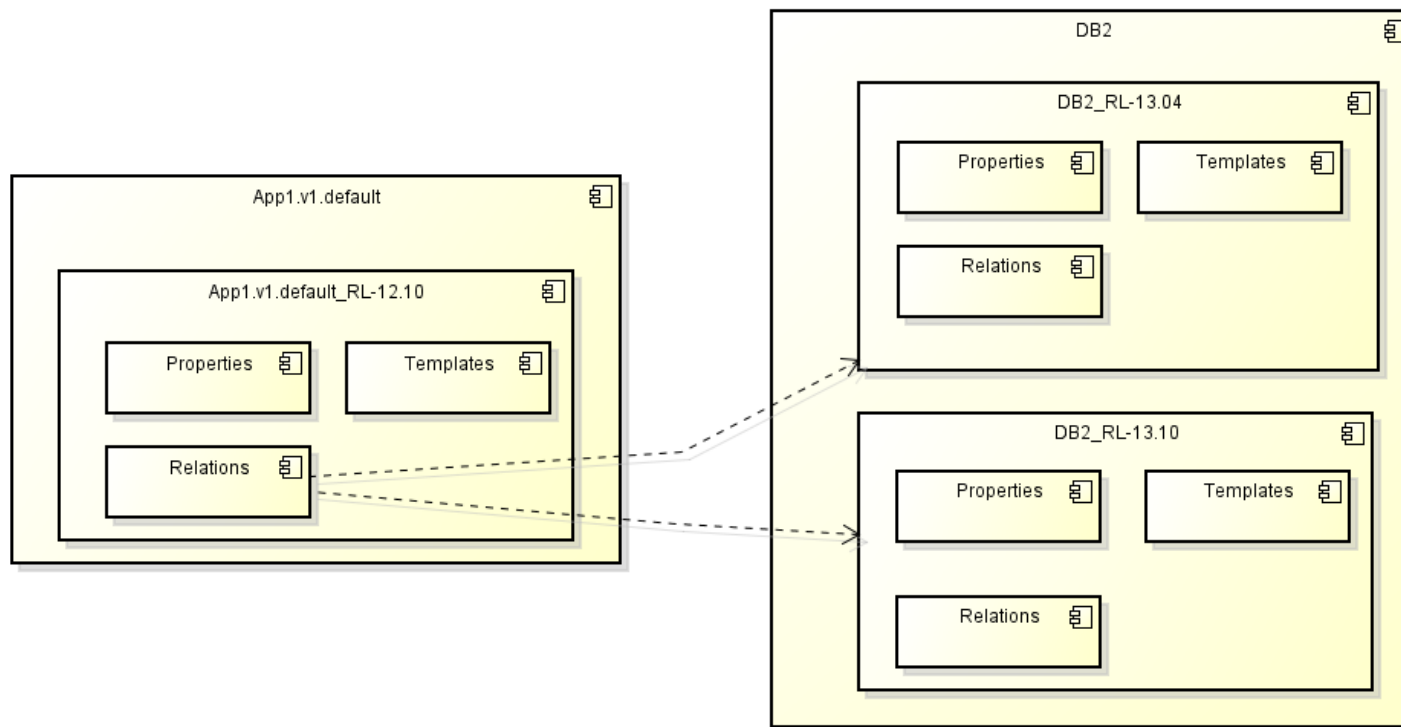


Releasing: Motivation

- In AMW ist die Konfiguration immer für alle Umgebungen gültig
- Auf den Entwicklungsumgebungen ist ein anderer Stand nötig, als auf Produktion
- Beispiel:
 - RL-15.04 wurde gerade produktiv gesetzt, nächster Release ist RL-15.10
 - Eine Applikation braucht im RL 15.04 eine andere Datenbank
 - Die Änderung kann nicht gemacht werden, da sonst Notfälle für RL-15.10 nicht mehr deployed werden können (DB noch nicht vorhanden)
 - Für Entwicklung ist die neue DB aber nötig
- Workarounds:
 - Ressource hinzufügen vor Deployment, dann wieder entfernen
 - Templates mit `#if` und `#else`
 - Config im AMW pro Umgebung umstellen

Releasing

- Release ist eine Kopie der Ressource: Properties, Templates, Relations
- Im AMW GUI als eine Element sichtbar
- Entspricht den BW und RL der Mobi
- Entspricht in etwa einer Branch im Sourcecode
- Alle Konsumenten haben automatisch auch eine Beziehung auf den neuen Release
- Bei Deployment wird die gültige Ressource ausgewählt

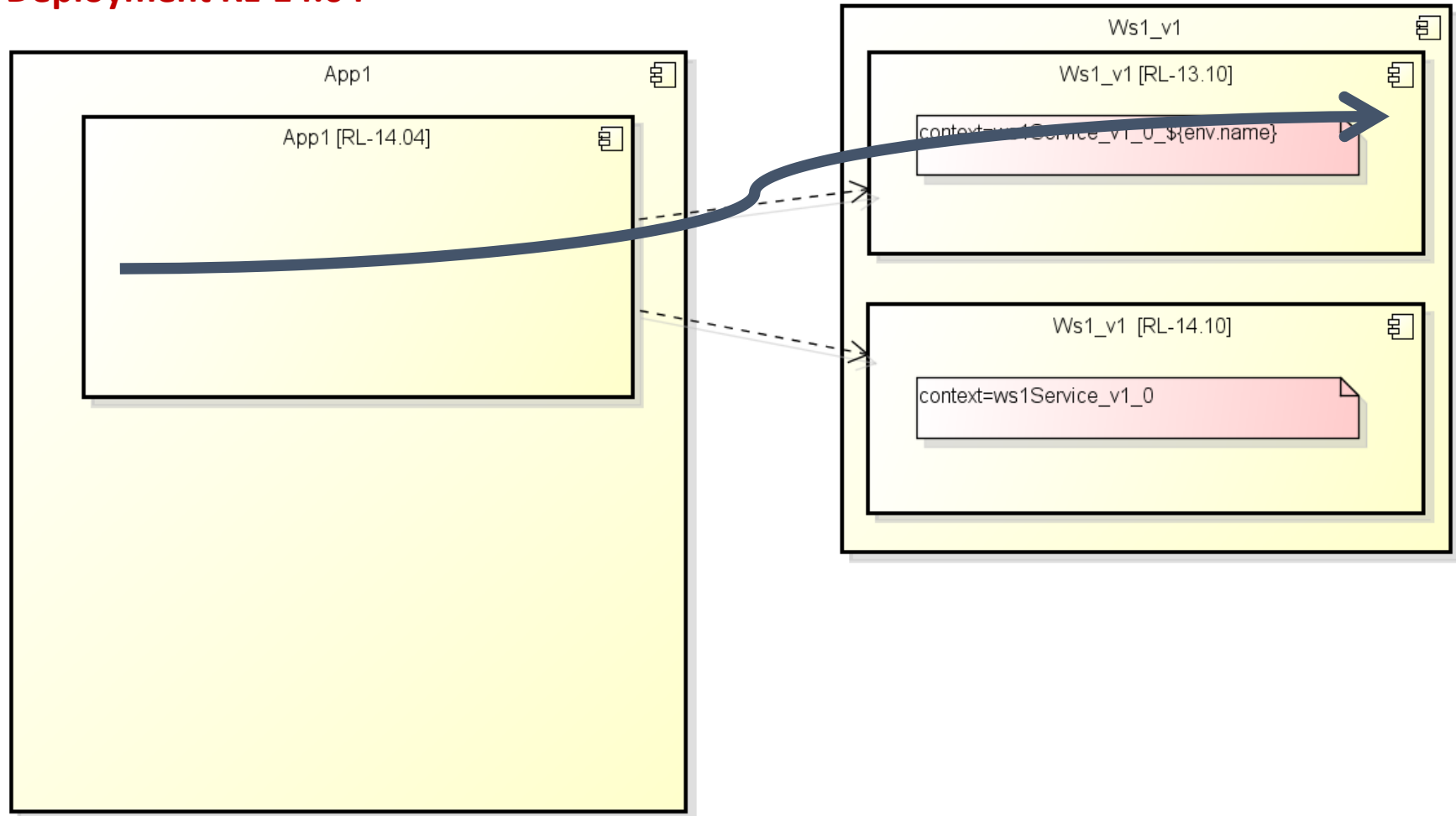


Releasing: Deployment

- Mit der Release Dropdown im Deploy Fenster kann der Pfad bestimmt werden, der gewählt wird.
- Verfügbare Einträge im Dropdown hängen vom Release des AppServers ab
- Applikationen können mit dem Release vom AppServer und höher deployed werden

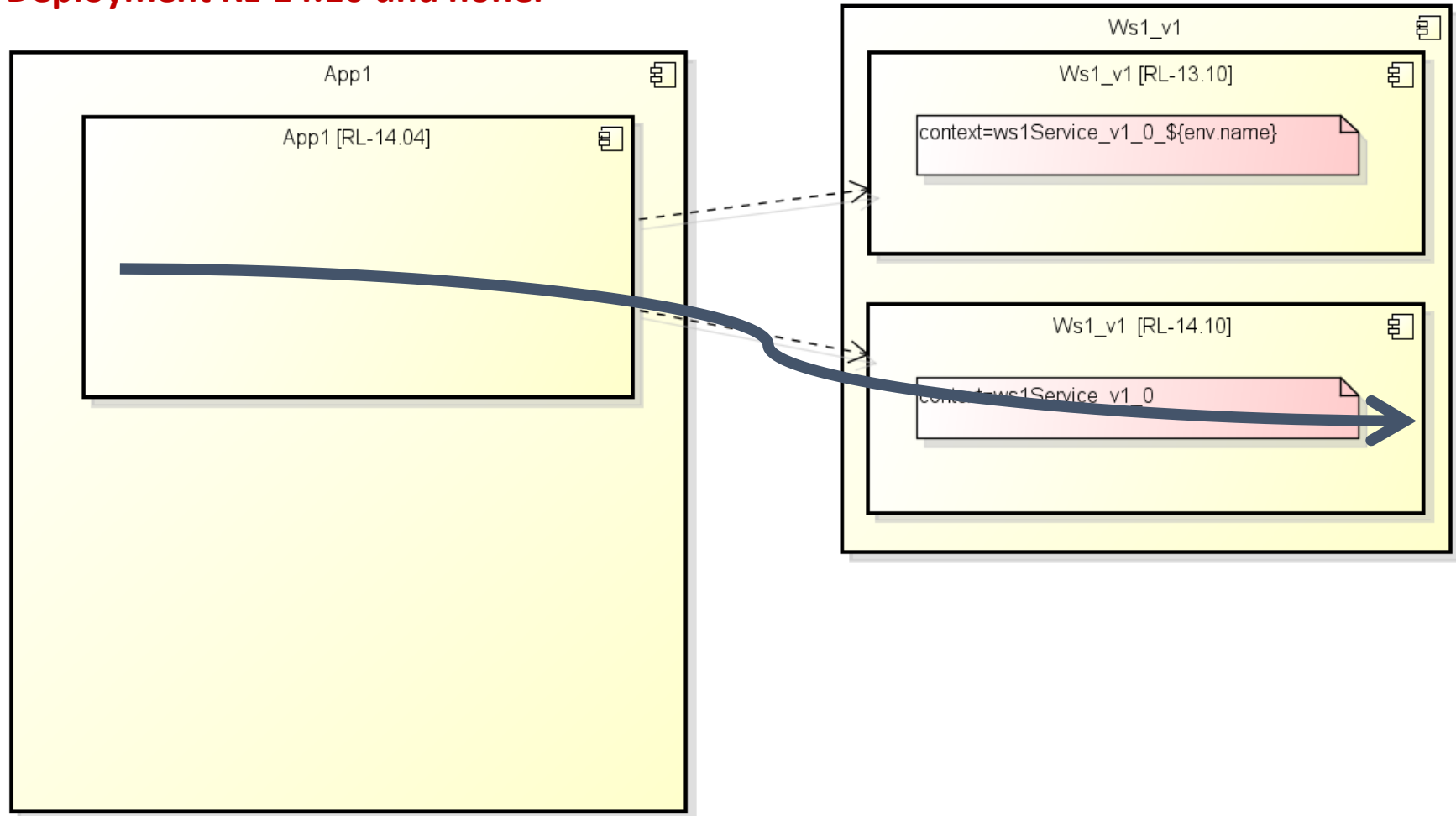
Modell: Releases

Deployment RL-14.04



Modell: Releases

Deployment RL-14.10 und höher

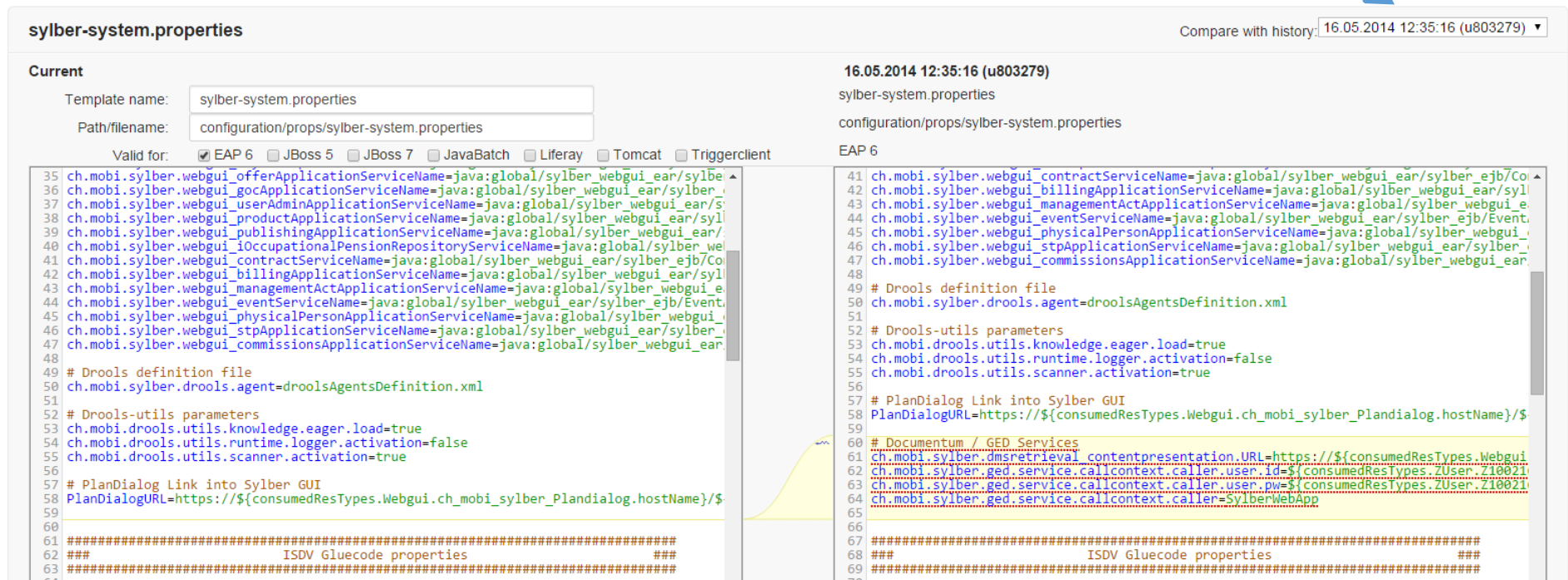


Template Editor

- JavaScript basierter Editor ([CodeMirror](#))
 - Eigene Suchfunktion mit Highlighting
 - Freemarker Syntax Highlighting
 - Zeilennummern
 - Fullscreen Modus
 - Wichtigste Tastenkombinationen sind unterhalb des Editors beschrieben
- Validierung von Freemarker Syntax beim Speichern
 - Erkennt nicht geschlossenen Klammern/Tags
 - Erkennt nicht, dass Variablen fehlen/nicht existieren! Dazu wird das Test Generate verwendet.

Template Editor: Vergleiche

- Ein Template kann mit alten Versionen verglichen werden
 - Editor zeigt die Unterschiede an
 - Unterstützt merging
 - Änderungszeitpunkt und User der die Änderung gemacht hat, wird im Dropdown angezeigt



The screenshot displays the 'sylber-system.properties' template editor. At the top right, a dropdown menu labeled 'Compare with history:' shows the selected version '16.05.2014 12:35:16 (u803279)'. Below this, the interface is split into two panels. The left panel, titled 'Current', shows the template name 'sylber-system.properties' and its path 'configuration/props/sylber-system.properties'. It also includes a 'Valid for:' section with checkboxes for various environments like EAP 6, JBoss 5, etc. The right panel, titled '16.05.2014 12:35:16 (u803279)', shows the same template details for the selected historical version. Both panels display the template content, which includes Drools definitions and parameters. A blue arrow points from the text 'wird im Dropdown angezeigt' to the 'Compare with history:' dropdown menu.

sylber-system.properties Compare with history: 16.05.2014 12:35:16 (u803279) ▼

Current

Template name: sylber-system.properties

Path/filename: configuration/props/sylber-system.properties

Valid for: ☒ EAP 6 ☐ JBoss 5 ☐ JBoss 7 ☐ JavaBatch ☐ Liferay ☐ Tomcat ☐ Triggerclient

```
35 ch.mobi.sylber.webgui_offerApplicationServiceName=java:global/sylber_webgui_ear/sylber_
36 ch.mobi.sylber.webgui_gocApplicationServiceName=java:global/sylber_webgui_ear/sylber_
37 ch.mobi.sylber.webgui_userAdminApplicationServiceName=java:global/sylber_webgui_ear/s
38 ch.mobi.sylber.webgui_productApplicationServiceName=java:global/sylber_webgui_ear/syl
39 ch.mobi.sylber.webgui_publishingApplicationServiceName=java:global/sylber_webgui_ear/
40 ch.mobi.sylber.webgui_iOccupationalPensionRepositoryServiceName=java:global/sylber_wel
41 ch.mobi.sylber.webgui_contractServiceName=java:global/sylber_webgui_ear/sylber_ejb/Co
42 ch.mobi.sylber.webgui_billingApplicationServiceName=java:global/sylber_webgui_ear/syll
43 ch.mobi.sylber.webgui_managementActApplicationServiceName=java:global/sylber_webgui_e
44 ch.mobi.sylber.webgui_eventServiceName=java:global/sylber_webgui_ear/sylber_ejb/Event
45 ch.mobi.sylber.webgui_physicalPersonApplicationServiceName=java:global/sylber_webgui_
46 ch.mobi.sylber.webgui_stpApplicationServiceName=java:global/sylber_webgui_ear/sylber_
47 ch.mobi.sylber.webgui_commissionsApplicationServiceName=java:global/sylber_webgui_ear
48
49 # Drools definition file
50 ch.mobi.sylber.drools.agent=droolsAgentsDefinition.xml
51
52 # Drools-utils parameters
53 ch.mobi.drools.utils.knowledge.eager.load=true
54 ch.mobi.drools.utils.runtime.logger.activation=false
55 ch.mobi.drools.utils.scanner.activation=true
56
57 # PlanDialog Link into Sylber GUI
58 PlanDialogURL=https://${consumedResTypes.Webgui.ch_mobi_sylber_Plndialog.hostName}/$
59
60
61 #####
62 ### ISDV Gluecode properties ###
63 #####
```

16.05.2014 12:35:16 (u803279)

sylber-system.properties

configuration/props/sylber-system.properties

EAP 6

```
41 ch.mobi.sylber.webgui_contractServiceName=java:global/sylber_webgui_ear/sylber_ejb/Co
42 ch.mobi.sylber.webgui_billingApplicationServiceName=java:global/sylber_webgui_ear/syl
43 ch.mobi.sylber.webgui_managementActApplicationServiceName=java:global/sylber_webgui_e
44 ch.mobi.sylber.webgui_eventServiceName=java:global/sylber_webgui_ear/sylber_ejb/Event
45 ch.mobi.sylber.webgui_physicalPersonApplicationServiceName=java:global/sylber_webgui_
46 ch.mobi.sylber.webgui_stpApplicationServiceName=java:global/sylber_webgui_ear/sylber_
47 ch.mobi.sylber.webgui_commissionsApplicationServiceName=java:global/sylber_webgui_ear
48
49 # Drools definition file
50 ch.mobi.sylber.drools.agent=droolsAgentsDefinition.xml
51
52 # Drools-utils parameters
53 ch.mobi.drools.utils.knowledge.eager.load=true
54 ch.mobi.drools.utils.runtime.logger.activation=false
55 ch.mobi.drools.utils.scanner.activation=true
56
57 # PlanDialog Link into Sylber GUI
58 PlanDialogURL=https://${consumedResTypes.Webgui.ch_mobi_sylber_Plndialog.hostName}/$
59
60
61 # Documentum / GED Services
62 ch.mobi.sylber.dmsretrieval.contentpresentation.URL=https://${consumedResTypes.Webgui
63 ch.mobi.sylber.ged.service.callcontext.caller.user.id=${consumedResTypes.ZUser.Z10021
64 ch.mobi.sylber.ged.service.callcontext.caller.user.pw=${consumedResTypes.ZUser.Z10021
65 ch.mobi.sylber.ged.service.callcontext.caller=SylberWebApp
66
67 #####
68 ### ISDV Gluecode properties ###
69 #####
```

Test Generate

- Über den Button „Test Generation“ auf den AMW Applicationservern erreichbar
- Zeigt weiterhin die Generierungsfehler der Templates an
- Neu werden auch die erstellten Templates angezeigt
 - Wird nur angezeigt, wenn der User Recht hat auf diese Umgebung zu deployen
- Releases können miteinander verglichen werden
 - Verwendet diff Funktion vom neuen Editor
 - Ab AMW 1.8 können Konfigurationsstände von einem bestimmten Datum miteinander verglichen werden

Test Generate

Release Selektion

The screenshot displays the 'Test Generate' interface. On the left is a vertical sidebar with environment categories: GLOBAL, DEV, B, C, LOCAL, N, S, TEAM (highlighted with a right-pointing arrow), U, V, W, X, Y, Z, INT, I, K, T, PRO, and P. The main area is titled 'Release Selektion' and shows a comparison between two releases: 'testAs1' (RL-14.10) and 'Compare with Release' (RL-15.10). Below this, the 'Application Server testAs1 on node_01' section lists templates. The 'Templates' section contains a table with two columns of template names, each preceded by a green checkmark. The templates are: amwScript/AmwRunScript.sh, amwScript/arguments.json, amwScript/jboss7_config.spec, amwScript/modelFromAppServer.tmp, amwScript/modelFromNode.tmp, amw_conf/configuration/run.conf, amw_conf/configuration/standalone.xml, and cmdb_export_testAs1_team_1.xml.tmp. The templates 'modelFromAppServer.tmp', 'modelFromNode.tmp', and 'cmdb_export_testAs1_team_1.xml.tmp' are highlighted in grey and have a double-headed arrow between them. Below the templates, the 'Applications' section shows a table with two columns of application names, each preceded by a green checkmark. The applications are: datasource_oracle.tmp, modelFromApp.tmp, and test. The 'test' application is highlighted in yellow and has a double-headed arrow between them. At the bottom, a code editor shows the test script: '1 test=jsplb0team.umobi.mobincorp.test' and '1 test=ssz.umobi.mobincorp.test'. Blue arrows point from the 'TEAM' sidebar item to the 'test' application, from the 'modelFromNode.tmp' template to the 'modelFromAppServer.tmp' template, and from the 'test=ssz.umobi.mobincorp.test' script to the 'test=jsplb0team.umobi.mobincorp.test' script.

Templates	Compare with Release
✓ amwScript/AmwRunScript.sh	✓ amwScript/AmwRunScript.sh
✓ amwScript/arguments.json	✓ amwScript/arguments.json
✓ amwScript/jboss7_config.spec	✓ amwScript/jboss7_config.spec
✓ amwScript/modelFromAppServer.tmp	✓ amwScript/modelFromAppServer.tmp
✓ amwScript/modelFromNode.tmp	✓ amwScript/modelFromNode.tmp
✓ amw_conf/configuration/run.conf	✓ amw_conf/configuration/run.conf
✓ amw_conf/configuration/standalone.xml	✓ amw_conf/configuration/standalone.xml
✓ cmdb_export_testAs1_team_1.xml.tmp	✓ cmdb_export_testAs1_team_1.xml.tmp

Applications	Compare with Release
✓ datasource_oracle.tmp	✓ datasource_oracle.tmp
✓ modelFromApp.tmp	✓ modelFromApp.tmp
✓ test	✓ test

```
1 test=jsplb0team.umobi.mobincorp.test
1 test=ssz.umobi.mobincorp.test
```

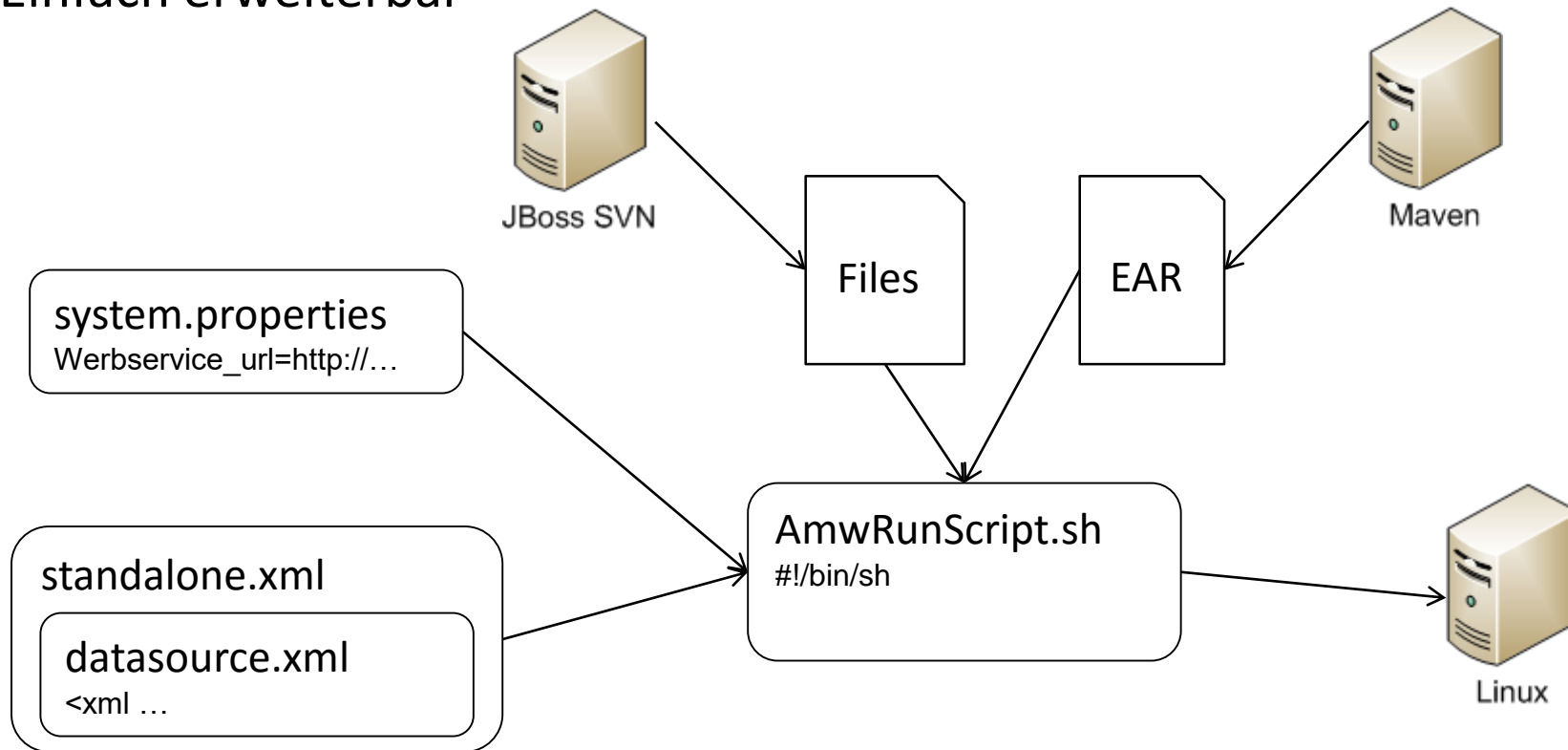
Hat es Template Fehler?

Gibt es überhaupt unterschiede?

Unterschiede der Templates

Deployment

- Paketieren der Templates und externen Files
- Installation auf Zielsystem
- Einfach erweiterbar



Deployment Filter

- Einzelne Filter werden automatisch mit AND oder OR verknüpft
- Deployment können als CSV exportiert werden.
- Latest deployment job for App Server and Env: nur letztes Deployment pro Umgebung

Infrastruktur Shakedown Test

- Testet die Ressourcen, die der Applikation angehängt sind
- Beispiel: prüfen ob Datenbankverbindung funktioniert
 - Firewall offen?
 - Treiber vorhanden?
 - DB vorhanden?
 - User und Password gültig?
- Tests werden auf das Zielsystem kopiert und dort ausgeführt
- Momentan sind zwei Tests für JBoss EAP implementiert:
 - Oracle
 - DB2
- Wiki:
https://mobiwiki.mobicorp.ch/wiki/index.php?title=AMW_Infrastruktur_Shakedown_Test

Best Practices

- Möglichst wenig Redundanzen
 - Zentrale Änderungen möglich
 - Übersicht
 - Props in Props: `${env.name?lower_case}`
- Ressourcen anstatt Properties
 - Wiederverwendung
 - Abhängigkeiten ersichtlich
- Von Anfang alle Umgebungen definieren
 - Konfig kann abgeschlossen werden
 - Weniger Fehler