inovat
innovative systeme verkehr tunnel technik

Technische Dokumentation

inoSystem
Entwicklungs- und Projektumgebung
Datenverteilersystem

Auftraggeber
inovat
innovative systeme – verkehr – tunnel – technik
Dipl.-Ing. H. C. Kniß
Siegstraße 31
D-50859 Köln

Ersteller
inovat
innovative systeme – verkehr – tunnel – technik
Dipl.-Ing. H. C. Kniß
Siegstraße 31
D-50859 Köln

Projekt
ia.inovat.06.01-inoSystem

Version: 2.2 Stand: 18.06.2007 Status: vorgelegt

1 Allgemeines

1.1 Verteiler

Organisationseinheit	Name	Anzahl Kopien	Vermerk
inovat	Dipl.-Ing. H. C. Kniß	1	

Tabelle 1-1: Verteiler

1.2 Änderungsübersicht

Version	Datum	Kapitel	Bemerkungen	Bearbeiter
1.0	06.03.2007	alle	Erstellung	Kniß (inovat)
2.0	11.03.2007	4.2	Ergänzung der Länder Saarland, Sachsen, Sachsen-Anhalt und Thüringen in der Ländertabelle	Kniß (inovat)
2.1	14.03.2007	3	Anpassung der Packagenamen an den neuen Präfix de.bsvrz	Kniß (inovat)
2.2	18.06.2007	alle	Diverse Fehlerkorrekturen, noch offene Punkte speziell markiert	Kniß (inovat)

Tabelle 1-2: Änderungsübersicht

1.3 Inhaltsverzeichnis

1.4 Abkürzungsverzeichnis

siehe [AbkBLAK]

1.5 Definitionen

Keine

1.6 Referenzierte Dokumente

VModell	V-Modell – Entwicklungsstandard für IT-Systeme (EstdIT), Version 1997.
DEnglisch	Dokument zur Festlegung der Verwendung deutscher und englischer Schlüsselwörter im Quellcode, Dokument *„Programmierung-Englische Schlüsselwörter.doc“*
NamenArc	Bezeichnungen der Segmente, SWE, Komponenten und Module gemäß Architektur mit Kurzbezeichnungen und daraus resultierenden Packagenamen, *„Bezeichnungen Architektur Datenverteilersystem.xls“*
KonfigKonventionen	Dokument mit Festlegungen zu Konventionen im Zusammenhang mit der Konfiguration des Datenverteilersystems *„Konventionen Konfiguraiton.doc“.“*

1.7 Abbildungsverzeichnis

1.8 Tabellenverzeichnis

2 Zweck des Dokuments

Das Dokument beschreibt die Vorgaben und Festlegungen für das Datenverteilersystem zu

- Konventionen zur Namensgebung von SWE / Modulen.
- Austausch von SWE und Modulen.
- Konventionen zur Namensgebung bei der Konfiguration.
- Austausch von Konfigurationen.
- Konventionen für Projektumgebungen[1]
- Austausch von Projektumgebungen

3 Konventionen zur Erstellung und Distribution von SWE und Modulen

3.1 Programmierung: Deutsch-Englisch

siehe [DEnglisch].

3.2 Bezeichnungen Packages

Die Bezeichnung von Packages bei der Entwicklung ist nach folgendem Schema zu bilden, wobei die Segment, SWE, Komponenten- und Modulbezeichnungen entsprechend der Architekturen (Systemarchitektur und Softwarearchitektur) zu bilden sind (Zusammenfassung siehe [NamenArc]).

`de.bsvrz.segment.swe.komponente{.[komponente|modul]*}.Klassenname`

mit

`de.bsvrz`	fester Präfix (bsvrz = **B**asis**S**ystem **VRZ**) für allgemein verfügbare SWE/Module des Datenverteilersystems, kann bei firmenspezifischen Entwicklungen natürlich ersetzt werden (z. B. durch **`de.inovat.nw.xxx`**).
`segment`	Segmentbezeichnung (Kurzform) gemäß Architektur (siehe [NamenArc]).
`swe`	SWE-Bezeichnung (Kurzform) gemäß Architektur. Folgende SWE-Kürzel sind derzeit vergeben: (siehe [NamenArc]).
`komponente`	Komponentenbezeichnung (Kurzform) gemäß Architektur (nur Kleinbuchstaben), (siehe [NamenArc]).
`modul`	Modulbezeichnung (Kurzform) gemäß Architektur, (siehe [NamenArc]).
`Klassenname`	Klassenname gemäß Javakonventionen (Erster Buchstabe groß), siehe jeweiliger Softwareentwurf.

[1] Im Dokumente gelb markierte Bereiche sind noch im Entwurfsstadium bzw. müssen noch ergänzt werden.

Die einzelnen Bestandteile des Packagenamens werden aus den Vorgaben wie folgt gebildet:

1. alle Buchstaben in Kleinbuchstaben wandeln
2. Umlaute `ä,ö,ü,ß` in `ae`, `oe`, `ue`, `ss` wandeln
3. alle sonstigen Sonderzeichen die nicht in der Zeichenklasse `[A-Za-z0-9]` enthalten sind, entfernen.

Besteht eine SWE nur aus einem Modul (welches häufig genauso heißt wie die SWE), so ist dieses Modul als eigenes Package aufzunehmen, auch wenn das Modul laut Architektur genauso heißt wie die SWE:

Beispiel SWE Güteberechnung (Guete) aus DUA, ein Modul Güteberechnung (Guete):

`de.bsvrz.dua.guete.guete`.[weitere Packages/Submodule laut SWEntw. oder Klasse]

Beispiel Segement (SEG), SWE (SWE), Komponente (KompA) mit Modulen (ModulA1 und ModulA2) sowie einem Modul (ModulB):

`de.bsvrz.seg.swe.kompa.modula1`.[weitere Packages/Submodule laut SWEntw. oder Klasse]

`de.bsvrz.seg.swe.kompa.modula2`.[weitere Packages/Submodule laut SWEntw. oder Klasse]

`de.bsvrz.seg.swe.modulb`.[weitere Packages/Submodule laut SWEntw. oder Klasse]

3.3 Distribution SWE und Module

Bei der Auslieferung (Austausch) werden immer einzelne SWE oder kleinere funktionale Einheiten (Komponente oder Module) zusammengestellt. Letzteres ist insbesondere bei Funktionbibliotheken mit einzelnen unabhängigen Modulen `xxx` und unterschiedlichen Herstellern z. B. aus der `de.bsvrz.sys.funclib.xxx`) oder SWE sinnvoll, bei der einzelne Module wie Plug-Ins entwickelt werden (z. B. OSI2 in den diversen KEx) oder bei denen einzelne Module von unterschiedlichen Herstellern gepflegt werden (z. B. `de.bsvrz.kex.tls.osi2` und `osi3` durch KS und `de.bsvrz.kex.tls.osi7` durch inovat).

Beispiel einer zu liefernden SWE `de.bsvrz.sys.funclib.cac`:

Geliefert wird ein (gezippter) Ordner mit dem PackageNamen der SWE bzw. des Moduls.

In diesem Ordner befinden sich mindestens folgende Dateien (fett gedruckte Namensanteile sind unabhängig vom PackageNamen):

de.bsvrz.sys.funclib.cac	Ordner
de.bsvrz.sys.funclib.cac**.jar**	jar-Datei mit den *.class-Dateien
de.bsvrz.sys.funclib.cac**-doc-api.zip**	Java-Doc (API, bis Level protected)
de.bsvrz.sys.funclib.cac**-doc-design.zip**	Java-Doc (API, bis Level private)
de.bsvrz.sys.funclib.cac**-src.zip**	Source, incl. Packagestruktur (de.bsvrz.sys. etc.)
de.bsvrz.sys.funclib.cac**-Build-Report.txt**	Infos zum Build (Datum, Version, Libs etc.)
de.bsvrz.sys.funclib.cac**-LGPL-lizenz.txt**	Lizenztext

Existieren (in analoger Packagestruktur) Testklassen (z. B. Junit), so werden diese analog zusammengefasst und sind ebenfalls im Ordner enthalten:

de.bsvrz.sys.funclib.cac**-test.jar**	jar-Datei mit den Test *.class-Dateien
de.bsvrz.sys.funclib.cac**-test-doc-api.zip**	wie oben, aber für Testklassen
de.bsvrz.sys.funclib.cac**-test-doc-design.zip**	wie oben, aber für Testklassen
de.bsvrz.sys.funclib.cac**-test-src.zip**	wie oben, aber für Testklassen
de.bsvrz.sys.funclib.cac**-test-Build-Report.txt**	optional für Testumgebung

Werden für die Ausführung weitere **EXTERNE** Bibliotheken benötigt (xerces etc.), so sind diese vollständig mit allen zugehörigen Dateien ebenfalls in den Ordner einzufügen (Namensschema der externen Bibliotheken kann natürlich vom Schema abweichen). Zusätzliche Bibliotheken sind jeweils in Unterordnern zuammenzufassen:

```
   LIB_A                                              Ordner mit allen Dateien der LIB A
      ZUSAETLICHE EXTERENE LIBS-A-lizenz.txt
      ZUSAETLICHE EXTERENE LIBS-A-weitere Dateien
      ZUSAETLICHE EXTERENE LIBS-A.jar
   LIB_B                                              Ordner mit allen Dateien der LIB B
      ZUSAETLICHE EXTERENE LIBS-B-lizenz.txt
      ZUSAETLICHE EXTERENE LIBS-B-weitere Dateien
      ZUSAETLICHE EXTERENE LIBS-B.jar
      etc.
```

Die Struktur der Sourcen entspricht direkt der Packagestruktur, d.h. beim Auspacken der `de.bsvrz.sys.funclib.cac`**`-src.zip`** erhält man folgende Sturktur:

```
de                                          Ordner
   bsvrz                                    Ordner
      sys                                   Ordner
         funclib                            Ordner
            cac                             Ordner
               eventuellWeiterePackages     Ordner (je nach SW-Arc)
                   *.java                   Java-Dateien des jeweiligen Packages
                   *.xyz                    sonstige Dateien des jeweiligen Packages
                   …                        (Bilder, ICONS, Property-Dateien etc.)
```

Für den xxx-Build-Report.txt ist folgendes Format einzuhalten:

Noch festzulegen: Ziel ist es, im Build-Report in einer automatisch auswertbaren Form alle notwendigen Informationen zu abhängigen SWE und deren Versions-/Buildinformationen abzulegen. Aus diesen Informationen sollen dann mittels eines noch zu erstellenden Werkzeugs automatisch die Abhängigkeiten zwischen den SWE dokumentiert werden.

Ein solcher Ordner kann einmalig in die jeweilige Entwicklungs- und Projektumgebung eingebunden werden. Ein Update beschränkt sich dann einfach auf den kompletten Austausch des Ordners.

4 Konventionen zur Erstellung und Distribution von Konfigurationsbereichen

4.1 Bezeichnung von Konfigurationsverantwortlichen / AOE

Konfigurationen werden unterschieden nach extern gelieferten Konfigurationsanteilen und nach Konfigurationsanteilen, die durch inovat gepflegt werden.

Die jeweiligen Konfigurationsverantwortlichen werden bei den durch inovat gepflegten Konfigurationsverantwortlichen nach folgendem Schema benannt:

`kv.[aoe.]`*`verantwortlicher`***`[.`***`ort`***`][.`***`auftraggeber`***`][.`** *`hierarchie`***`]`**

mit

kv festes Präfix für **K**onfigurations**v**erantwortlicher.

aoe	zusätzliches festes Präfix für **A**utarke**O**rganisations**E**inheit, wenn Konfigurationsverantwortlicher auch autarke Organisationseiheit ist. Dies sollte normalerweise nur bei Konfigurationsverantwortlichen der Fall sein, die zum Ausführen einer Konfiguration in einem konkreten Projekt verwendet werden. Allgemeine Konfigurationsbereiche mit Modellen, die von anderen Anwendern verwendet werden, sollten ausschließlich als einfacher Konfigurationsveantwortlicher gepflegt und publiziert werden
verantwortlicher	Kürzel des Verantwortlichen, Firma (bea, bitctlr, dambach, inovat, kappich ,logos, etc.) bzw. Land , siehe Kapitel 4.2.
ort	optional, Sitz des Auftraggebers und/oder Ortsangabe für die zusätzlich angegebene **hierachie**, siehe Beschreibung Kapitel 4.2.
auftraggeber	optional, siehe Beschreibung bei Kapitel 4.2.
hierachie	optionale weitere Unterscheidung (fettgedruckt), wie z. B.: o kv.aoe.inovat.nw.stnrw**.vrz.leverkusen** o kv.aoe.inovat.nw.stnrw**.uz.loehne** o kv.aoe.inovat.nw.stnrw**.uz.weilerswist** o kv.aoe.inovat.nw.boschung**.zentrale.kamen**

Die optionalen Zusätze **ort, auftraggeber,** und **hierachie** werden i. d. R. durch den Auftraggeber zentral verwaltet.

Die Konfigurationsbereiche (mit Endung `*.xml` bzw. `*.config`) sowie die Ordnernamen entsprechen der Bezeichnung des Konfigurationsverantwortlichen, also z. B.:

Konfigurationsverantwortlicher / AOE	**Bemerkung**
kv.bea	Allgemeine Modelle, durch beck et al gepflegt.
kv.bitctrl	Allgemeine Modelle, durch BitCtrl gepflegt.
kv.dambach	Allgemeine Modelle, durch Dambach gepflegt.
kv.inovat	Allgemeine Modelle, durch inovat gepflegt.
kv.kappich	Allgemeine Modelle, durch Kappich Systemberatung gepflegt
kv.logos	Allgemeine Modelle, durch Logos gepflegt
kv.bw	Allgemeine Modelle, durch Baden-Württemberg gepflegt
kv.nw	Allgemeine Modelle, durch Nordrhein-Westfalen gepflegt
…	…weitere…
kv.inovat.nw	Allgemeine Modelle für NRW, durch inovat gepflegt.
…	…weitere…
kv.**aoe.inovat**.nw.stnrw**.vrz.leverkusen**	**AOE** für den Betrieb der speziellen Versorgung der VRZ Leverkusen in NRW, Verantwortlich **inovat,** Ort NRW und Auftraggeber StraßenNRW.
kv.**aoe.inovat**.nw.boschung**.uz.kamen**	**AOE** für den Betrieb der speziellen Versorgung der Boschung Zentrale in Kamen in NRW, Verantwortlich **inovat,** Ort NRW und Auftraggeber Boschung Mecatronic.
kv.**aoe.nw.vrz.leverkusen**	**AOE** für den Betrieb der speziellen Versorgung der VRZ Leverkusen in NRW, Verantwortlich **NW. Ort und Aufttraggeber wurden weggelassen, da identisch mit Verantwortlichem**.

Tabelle 4-1: Beispiele für Konfigurationsverantwortliche /AOE

4.2 Länderkürzel

Land bzw. Bundesland, in dem der Auftraggeber sitzt. In bestimmten Ausnahmefällen wird hier auch der Ort indirekten Auftraggebers verwendet, wenn z. B. eine Behörde ein Unternehmen beauftragt, welches diesen Auftrag als Unterauftrag weitergibt.

Orts-/Länderkürzel	**Land**
bb	Brandenburg
be	Berlin

Orts-/Länderkürzel	Land
bw	Baden-Württemberg
by	Bayern
hb	Bremen
he	Hessen
hh	Hamburg
mv	Mecklenburg-Vorpommern
ni	Niedersachsen
nw	Nordrhein-Westfalen
rp	Rheinland-Pfalz
sh	Schleswig-Holstein
sl	Saarland
sn	Sachsen
st	Sachsen-Anhalt
th	Thüringen
de	Deutschlandallgemein
ch	Schweiz
at	Österreich
ia	interne Projekte

Tabelle 4-2: Länderkürzel für Ortsbezüge

4.3 Bezeichnung von Konfigurationsbereichen

Konfigurationsbereiche sollten folgenden Namenskonventionen folgen:

kb.*KV_AOE*	Für den Konfigurationsverantwortlichen/AOE
kb.default.*KV_AOE*	Für den Defaultbereich des KV/AOE
kb.tm*Name*	Für Teilmodelle
kb.objekte*Name*	Für Objektversorgungen

mit

kb	festes Präfix für **K**onfigurations**B**ereich.
default	zusätzliches festes Präfix für den **Default**konfigurationsbereich des zuständigen KV/AOE.
KV_AOE	KV/AOE entsprechend Vorgaben Kapitel 4.1.
tm	zusätzliches festes Präfix für **T**eil**M**odell.
objekte	zusätzliches festes Präfix für Konfigurationsbereiche mit konkreten **Objekte**definitionen.
Name	Name des Modells bzw. Bereichs.

Beispiele für KV/AOE: `kv.aoe.nw.vrz.leverkusen`

- **kb.**kv.aoe.nw.vrz.leverkusen
- **kb.default.**kv.aoe.nw.vrz.leverkusen

Beispiele für Teilmodell bzw. Objektversorgungen

- **kb.tm**FachModellGlobal
- **kb.objekte**NwUzDuerenVerkehrMqFs

4.4 Distribution Konfigurationsbereiche

Von (für) einen Konfigurationsverantwortlichen bzw. AOE werden immer alle Konfigurationsdateien geliefert. Die Struktur der Lieferung wird nachfolgend am Beispiel erläutert.

Beispiel einer zu liefernden Konfiguration für einen KV/AOE: **kv.inovat**, der für folgende Konfigurationsbereiche zuständige ist:

- kb.kv.inovat
- kb.tmEreignisKalenderGlobal
- kb.tmGanglinienGlobal
- kb.tmGeoReferenzierungGlobal
- kb.tmKExTlsGlobal
- kb.tmSystemKalenderGlobal
- kb.tmTmcGlobal
- kb.tmUmfeldDatenGlobal
- kb.tmVerkehrGlobal

Geliefert wird ein (gezippter) Ordner mit dem Namen des KV/AOE.

In diesem Ordner befinden sich mindestens folgende Dateien (fett):

```
kv.inovat                                          Ordner
      config                                       Unterordner mit *.config und
                                                   Verwaltungdatei verwaltung.xml

            kb.kv.inovat.config                    Config-Dateien des Verantwortlichen
            kb.tmEreignisKalenderGlobal.config
            kb.tmGanglinienGlobal.config
            kb.tmGeoReferenzierungGlobal.config
            kb.tmKExTlsGlobal.config
            kb.tmSystemKalenderGlobal.config
            kb.tmTmcGlobal.config
            kb.tmUmfeldDatenGlobal.config
            kb.tmVerkehrGlobal.config
            verwaltungsdaten.xml                   Verwaltungsdatei des Verantwortlichen

      xml                                          Ordner mit Export-Dateien)

            kb.kv.inovat.xml                       Exportierte XML Datei, optional
            kb.tmEreignisKalenderGlobal.xml
            kb.tmGanglinienGlobal.xml
            kb.tmGeoReferenzierungGlobal.xml
            kb.tmKExTlsGlobal.xml
            kb.tmSystemKalenderGlobal.xml
            kb.tmTmcGlobal.xml
```

```
            kb.tmUmfeldDatenGlobal.xml
            kb.tmVerkehrGlobal.xml
      ReleaseNotes                                  Ordner mit Änderungeinformationen

            Datenmodelle-kv.inovat.pdf              Optional: Zusätzliche Dokumentation
            ReleaseNotes kv.inovat.doc              Änderungsverfolgung für diesen KV
```

Die `verwaltung.xml`, die zur Erzeugung verwendet wurde, ist mitzuliefern. Aus dieser lässt sich u. a. erkennen, welche zusätzlichen Abhängigkeiten bei der Verwendung der Konfigurationsbereiche des KV/AOE zu anderen KV/AOE bestehen! Im Beispiel sieht man die Abhängigkeit zu `kv.kappich` und `kv.bea`

Die *,xml Dateien sind mitzuliefern, um einen zusätzlichen Exportvorgang auf der Empfängerseite überflüssig zu machen.

Der Ordner ReleaseNotes ist mitzuliefern. Diese enthält mindestens das Dokument **ReleaseNotes kv.xxx.doc** mit den Änderungsinformationen zu jedem Konfigurationsbereich je Version des Bereichs für den gelieferten Konfigurationsverantwortlichen. Eine Beispieldatei für dieses Dokument kann dem kv.inovat entnommen werden. Das Dokument darf ausdrücklich für den jeweiligen KV angepasst werden (LOGO, Verantwortlicher etc.).

Im Ordner ReleaseNotes sollten zudem zusätzlich Informationen zum KV mitgeliefert werden (z. B. grafische Darstellung der Modelle etc.).

Für das obige Beispiel sieht die `verwaltung.xml` wie folgt aus. Die Pfadangaben zu den zusätzlich benötigten Konfigurationsdateien sind relativ zum KV/AOE-Ordner (hier also zu kv.inovat), wobei die Dateien der anderen Konfigurationsverantwortlichen ebenfalls in Ordner stehen, die genau der hier beschriebenen Konvention entsprechen.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE verwaltungsdaten PUBLIC "-//K2S//DTD Verwaltung//DE" "verwaltungsdaten.dtd">
<verwaltungsdaten>
   <konfigurationsverantwortlicher pid="kv.inovat"/>
   <konfigurationsbereich pid="kb.kv.inovat" verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmEreignisKalenderGlobal"
                          verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
      <version nr="2" zeitpunkt="05.03.2007 20:54:15,698"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmGanglinienGlobal" verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
      <version nr="2" zeitpunkt="05.03.2007 20:54:15,698"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmGeoReferenzierungGlobal"
                          verzeicnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmKExTlsGlobal" verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
      <version nr="2" zeitpunkt="05.03.2007 20:54:15,698"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmSystemKalenderGlobal"
                          verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmTmcGlobal" verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
```

```
   <konfigurationsbereich pid="kb.tmUmfeldDatenGlobal" verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmVerkehrGlobal" verzeichnis="../../kv.inovat/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
      <version nr="2" zeitpunkt="05.03.2007 20:54:15,698"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.kv.bea" verzeichnis="../../kv.bea/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmVewProtokolleGlobal" verzeichnis="../../kv.bea/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.fachModellGlobal" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
      <version nr="2" zeitpunkt="05.03.2007 20:54:15,698"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.kv.kappich" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.metaModellGlobal" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.systemModellAoe" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.systemModellGlobal" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmIlseTls" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmVewBetriebGlobal" verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
   <konfigurationsbereich pid="kb.tmVewSimulationGlobal"
                          verzeichnis="../../kv.kappich/config/">
      <version nr="1" zeitpunkt="03.03.2007 17:58:38,000"/>
   </konfigurationsbereich>
</verwaltungsdaten>
```

4.5 Fehlersuche – Log- und Debugausgaben

Siehe dazu [KonfigKonventionen].

4.6 Konventionen zur Vergabe von PID's

Siehe dazu [KonfigKonventionen].

4.7 Konventionen zur Aufteilung und Verwaltung von Konfigurationsbereichen

Siehe dazu [KonfigKonventionen].

5 Konventionen zur Erstellung und Distribution von Projekten

Die zuvor beschriebenen Richtlinien für die Erstellung und Distribution von SWE und Konfigurationsbereichen stellen die Grundlage für die Distribution von Projektumgebungen dar. Projekte sind konkrete Zusammenstellungen von SWE, Konfigurationsbereichen, Parametersätzen, Start- und Stoppskripten etc. für ein konkretes Projekt. Obwohl konkrete Projektumgebungen i. d. R. nicht global ausgetauscht werden, macht eine Konvention zur Erstellung und Distribution von Projekten aus folgenden Gründen Sinn:

- Einarbeitungszeit, Verständnis und Unterstützungsmöglichkeiten werden vereinfacht, da alle Projekte strukturell identisch aufgebaut sind.
- Beispielprojekte, die den Einsatz oder die beispielhafte Verwendung von SWE demonstrieren, sind für alle direkt verständlich und nachvollziehbar.
- Aufgrund der identischen Projektstrukturen (Verzeichnisstruktur) können Start- /Stoppskripte aus Beispielprojekten direkt in eigenen Projekten verwendet werden.
- Projektupdates realer Projektumgebungen können vereinheitlicht und (weitestgehend) automatisiert werden.

Denkar ist z. B. folgende Struktur, die auf die Richtinien für die Erstellung und Distribution von SWE und Konfigurationsbereichen aufbaut:

```
└──Projektname
   ├──archiv
   ├──bin
   │  ├──ant
   │  ├──bash
   │  ├──dos
   │  └──logs
   ├──debug
   ├──dokumentation
   ├──konfiguration
   │  ├──config
   │  ├──log
   │  └──xml
   ├──lib
   │  ├──de.bsvrz.ars.ars
   │  ├──de.bsvrz.ars.export
   │  ├──de.bsvrz.kex.centromapvrz
   │  ├──de.bsvrz.kex.lcl
   │  ├──de.bsvrz.pat.inovatkonfigtool
   │  ├──de.bsvrz.pua.prot
   │  ├──de.bsvrz.sys.funclib.csv
   │  ├──de.bsvrz.sys.funclib.debug
   │  ├──de.bsvrz.sys.funclib.kbgen
   │  ├──jre1.6.0
   │  └──zz.ks.kernsystem
   ├──log
   └──parameter
```

- Im Ordner `archiv` befinden sich die Archivdaten für das Projekt.
- Im Ordner `bin` befinden sich Unterordner mit den (Start-/Stopp)Skripten für das Projekt.
- Im Ordner `konfiguration` befinden sich Unterordner mit den Konfigurations- und XML-Dateien für das Projekt.

- Im Ordner `lib` befinden sich die Unterordner mit den SWE (wie unter „Distribution von SWE“ beschrieben)
- Im Ordner `parameter` befinden sich die Parameterdaten für das Projekt.