



Software-Praktikum

Sitzung zum Meilenstein 2

- Korrektur der zu MS2 gefundenen Fehler
- Generelle Anmerkungen zu den Abgaben zu MS2
- Hinweise zu MS3
- Fragen
- Treffen mit ausgewählten Teilnehmern

Korrektur der zu MS2 gefundenen Fehler bis MS3

- Fehlermeldungen kommen noch bis zum 20.05.
 - Jeder korrigiert die im Gitlab Issue Tracker dokumentierten Fehler und die mündlich angesprochenen Fehler seines Arbeitspaketes.
 - Fehler haben Label `bug`
 - Empfehlungen haben Label `suggestion`
 - Die Zustände der im Gitlab Issue Tracker dokumentierten Fehler müssen Sie korrekt setzen, um die Korrektur für uns beobachtbar zu machen, d.h.
 - Sie setzen nach vorgenommener Fehlerkorrektur und push des Codes den Zustand/Label des Issues auf „`closed`“.
 - Wir kontrollieren Ihre Fehlerkorrektur und setzen den Zustand/Label des Issues auf
 - „`approved`“, falls Ihre Korrektur von uns akzeptiert wird,
 - „`reopened`“, falls Ihre Korrektur nicht ausreichen war.
- > Diesen Issue müssen Sie dann erneut korrigieren.

Generelle Anmerkungen zu den Abgaben zu MS2

- **IDE Warnungen** beachten (hätte viele Bugs/Suggestions vermieden)
- Beim **Mergen** der lokalen Änderungen mit den Änderungen im Remote Repository aufpassen: Keine Abgaben anderer Teilnehmer löschen!
- Vordefinierte **NamedQuery** verwenden statt eigene Query erstellen: NamedQuery findet man z.B. an den Entity Klassen im wawi-dbmodel in Gitlab)
- **Package-Namen** erstellen wie in 5.3 in Vorgaben beschrieben
- **Pipeline Status** im Gitlab beachten: Es gab mehrere Teilnehmer, bei denen die letzten Pipelines nicht mehr durchgelaufen sind
- **Aussagekräftige** Parameter- und Variablennamen wählen (wie in SE gelernt)
 - Die Parameternamen, die IntelliJ generiert sind schlecht!

Generelle Anmerkungen zu den Abgaben zu MS2

- **Exceptions:** wenn Exceptions geworfen werden, dann stimmt was nicht. Meist die `DetachedEntityException`: ein Objekt mit einem `NestedObject` soll in die DB eingefügt werden, aber das `NestedObject` ist der DB nicht bekannt und wird auch nicht kaskadierend eingefügt.
- Methoden nicht unnötig kompliziert machen: man muss nicht jede Variable überall auf `null` prüfen, z.B. bei jedem Insert alle Attribute einzeln durchlaufen ob die besetzt sind
- Architektur beachten: wir haben die Schnittstelle für die Datenhaltung mit der **DatenhaltungAPI** vorgegeben, diese darf nicht selber innerhalb der eigenen Komponenten neu erzeugt werden.

Generelle Anmerkungen zu den Abgaben zu MS2

- unnötiger Code = falsche Code ;-)
Dinge wie:

```
if(kunde == null) {  
    return null;  
}  
return kunde;
```

oder

```
if(lagerverkehr == null) {  
    return null;  
}  
else{  
    return lagerverkehr;  
}
```

sollten grundsätzlich entfernt werden.

Generelle Anmerkungen zu den Abgaben zu MS2

- überall typisieren, wo es möglich ist, also z.B. :

```
List<Lagerverkehr> lagerverkehr =  
entityManager.createNamedQuery("Lagerverkehr.findAll",  
Lagerverkehr.class).getResultList();
```

statt

```
List<Lagerverkehr> lagerverkehr =  
entityManager.createNamedQuery("Lagerverkehr.findAll").getResult  
tList();
```

und natürlich immer

TypedQuery<Kunde> = ... statt Query =...

bzw.

List<Kunde> kunden = ... statt List kunden =...

Generelle Anmerkungen zu den Abgaben zu MS2

- Parameter in TypedQuery's per `setParameter`-Methode setzen, statt in den String rein zu verketten, also

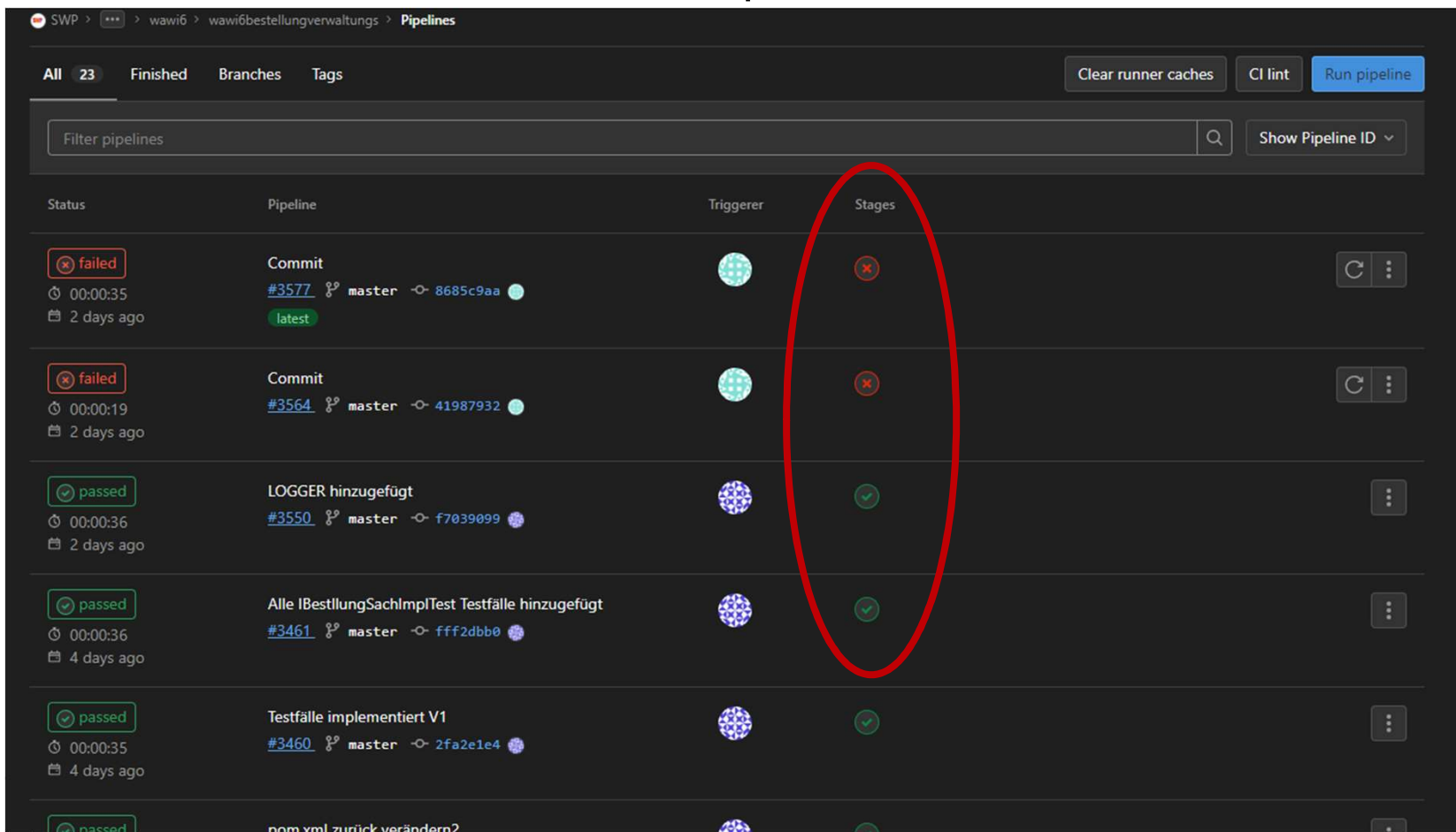
```
TypedQuery<Lagerverkehr> typedQuery =
entityManager.createQuery("SELECT l FROM Lagerverkehr l
WHERE created BETWEEN :param1 and :param2",
Lagerverkehr.class);
typedQuery.setParameter("param1", localDate);
typedQuery.setParameter("param2", localDate1);
```

statt

```
TypedQuery<Lagerverkehr> typedQuery =
entityManager.createQuery("SELECT l FROM Lagerverkehr
l WHERE created BETWEEN " + localDate.toString() + "
and " + localDate1.toString()", Lagerverkehr.class);
```

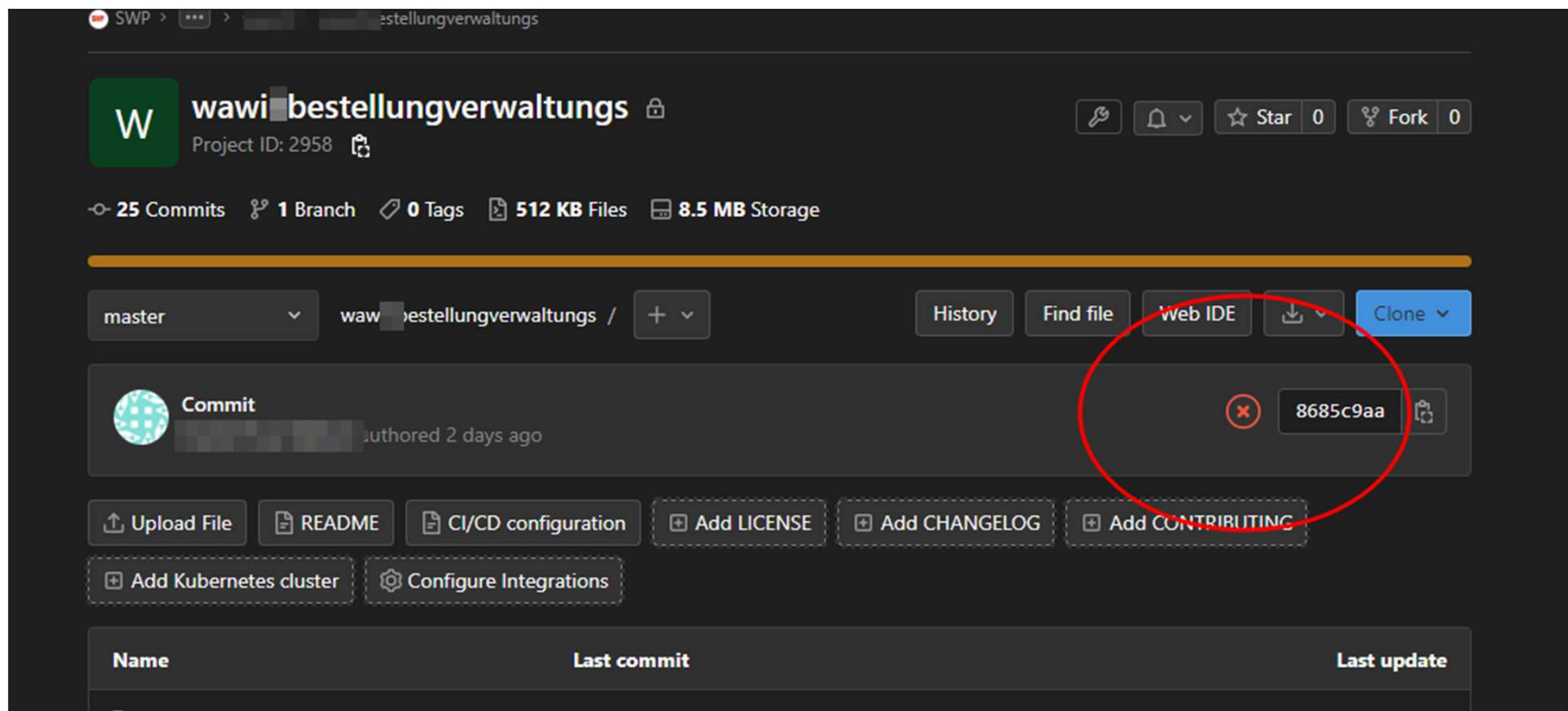
Die CI/CD-Pipeline im SWP

- Pipeline-Seite des Projekts in Gitlab beachten:
 - Projekt/Komponente in Gitlab
 - Linke Menüleiste: CI/CD > Pipelines



Die CI/CD-Pipeline im SWP

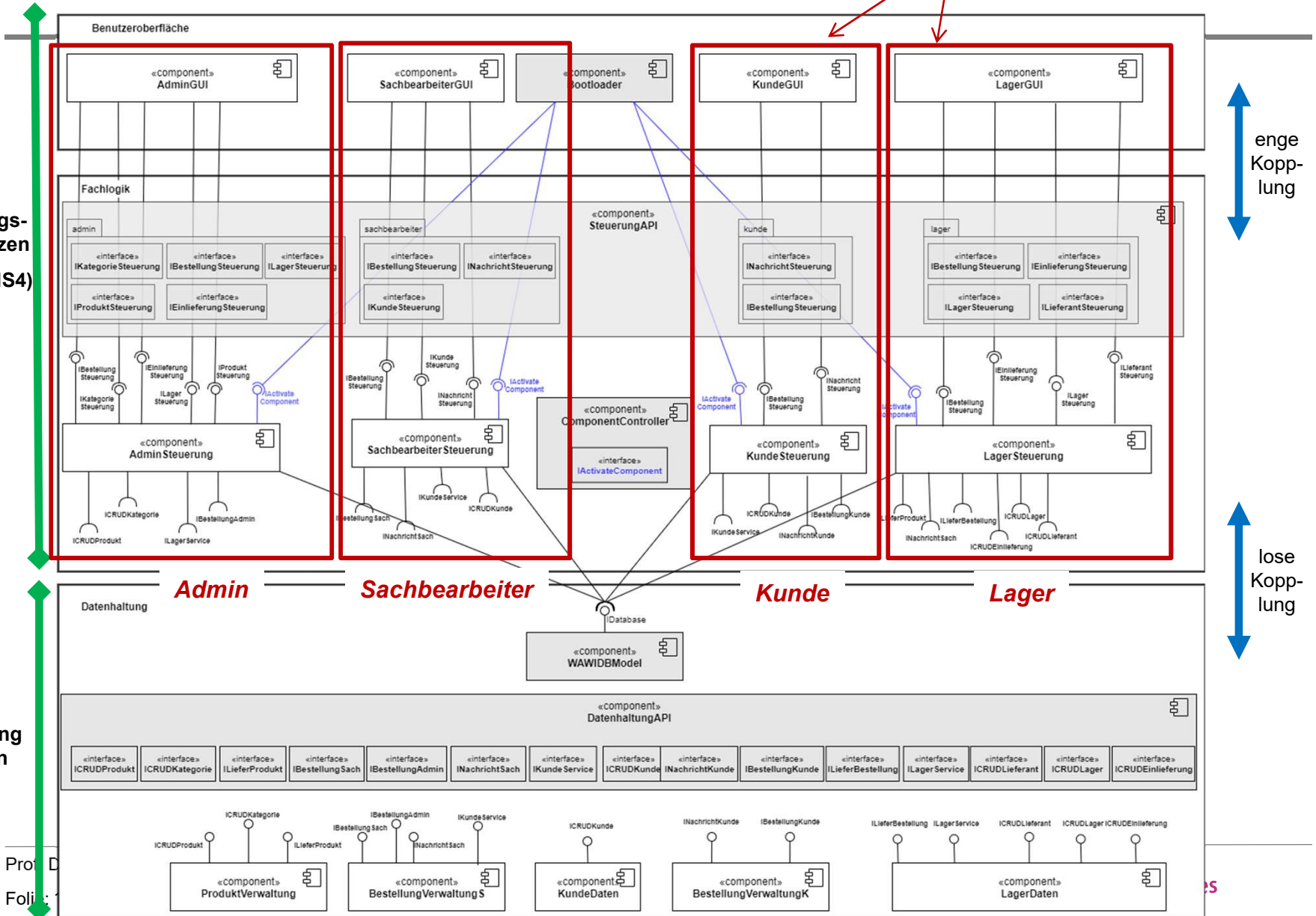
- Man kann den Erfolg auch immer auf der Einstiegsseite vom Projekt sehen:
 - Ist da ein rotes X, dann ist die letzte Pipeline nicht erfolgreich durchgelaufen.



Hinweise zu MS3

vertikale fachliche Strukturierung

Phase 3:
Anwendungs-
fälle umsetzen
(MS3 und MS4)



Hinweise zu MS3

- Sie sind alle in Teams eingeteilt worden
 - In diesem Team bearbeiten Sie die Meilensteine 3, 4 und 5
 - Tauschen Sie Kontaktdaten in Ihrem Team aus
 - Insbesondere Telefonnummern
 - Emailadressen sind in Ilias
 - Sie haben die Arbeitspakete für MS3 und MS4 erhalten
- MS3: Klassendiagramm, Anwendungsfall-Tabelle, GUI-Entwurf
 - Ziel: sehr gutes Klassendiagramm als Basis für die Implementierung
 - Klassendiagramm für Schicht „Fachlogik“, nicht für Schicht „Benutzeroberfläche“
 - Bezüglich Datenschicht nur Zugriffe auf deren Interfaces darstellen
 - Alle Dokumente werden im Team gemeinsam erstellt
 - Vorlage für Anwendungsfall-Tabelle auf Ilias
 - Klassendiagramm muss während gesamtem Projekt aktuell gehalten werden
 - GUI-Entwurf = Wireframes oder Bilders eines GUI-Builders oder Zeichnungen der geplanten Oberfläche (in PowerPoint o.ä.) der einzelnen Anwendungsfälle







Hinweise zu MS3:

Anwendungsfall-Tabelle

Anwendungsfall-Tabelle

System: *Name des Systems (z.B. WAWI1)*

Komponente: AdminSteuerung

- Anwendungsfall „LF100/ Produkte anlegen, bearbeiten, löschen“
 - Interface-Klasse: 
 - Methode der Interface-Klasse: 
- Anwendungsfall „/LF105/ Produkt-Kategorien anlegen, bearbeiten, löschen“
 - Interface-Klasse: 
 - Methode der Interface-Klasse: 
- Anwendungsfall „/LF140/ Liste aller Bestellungen anzeigen“
 - Interface-Klasse: 
 - Methode der Interface-Klasse: 
- Anwendungsfall „/LF110/ Alle Produkte mit Kategorien anzeigen, Produkte aktivieren und deaktivieren“
 - Interface-Klasse: 
 - Methode der Interface-Klasse: 

Hinweise zu MS3

- **Liefergegenstände:**
 1. Klassendiagramm der Fachlogik im pdf-Format
 2. Anwendungsfall-Tabellen im pdf-Format
 3. GUI-Entwurf in allgemein lesbarem Format (pdf, html, etc.), so dass wir kein spezielles Tool bei uns installieren müssen
 4. Alle Bugs sind korrigiert und im Gitlab Issue Tracker mit korrektem Status gesetzt
- **Abgabe aller Liefergegenstände bis zum 31.05., 8:00 Uhr morgens:**
 - im Git Remote Repository, im Branch master, gekennzeichnet mit Tag MS3
 - für Klassendiagramm und Anwendungsfall-Tabellen: Verzeichnis docs in der entsprechenden Komponente der Fachlogik-Schicht
 - für GUI-Entwurf: Verzeichnis docs in der entsprechenden Komponente der Benutzeroberfläche-Schicht
 - Das Verzeichnis docs müssen Sie selbst anlegen, falls es noch nicht vorhanden ist.

Hinweise zu MS3

- Sitzung zum Meilenstein: 03.06., 13:00 Uhr, Zoom
 - Vorbereitung auf MS4
 - Einzelgespräche mit einigen Teams im Anschluss an gemeinsamer Besprechung
- GUI-Entwurf: Separate Online-Termine für jedes Team zwischen 03.06. und 10.06.:
 - Auf Treffen spielen die Teams die Aktionen der Anwendungsfälle mit Hilfe ihres GUI-Entwurfs in einer Präsentation für die Betreuer durch
- Klassendiagramm und Anwendungsfall-Tabellen: Kommentare zu von uns bis Freitag, 10.06.
 - Innerhalb neuer Version des pdf-Dokuments

Fragen

- Bitte stellen Sie Ihre Fragen!
 - zu MS3
 - zu MS4
 - zu MS5
 - zum SWP
 - ...