Anmerkung: Grundsätzlich sollten für die SEPM-Prüfungsvorbereitung (Biffl) etwa 3-4 Tage einkalkuliert werden. Wenn man schon einige Übung im Modellieren von UML-Diagrammen (Klassen-, Aktivitäts- und Zustandsdiagramm) hat, reicht es im Normalfall aus, die grundlegenden Techniken nochmal zu wiederholen und evtl. eine Aufgabe durchzumodellieren. Sollte man die grundlegenden Elemente der jeweiligen UML-Diagramme nicht mehr kennen, würde ich empfehlen mindestens einen Tag mehr einplanen.

Die folgenden Fragen decken die Prüfungen der letzten Jahre (zumindest die, die im VoWi verlinkt wurden) zur Gänze ab. Ich habe die Prüfung am 21.01.2013 geschrieben und wirklich überrascht wurde ich nicht. Es kamen exakt dieselben Fragen (zum Teil sogar die gleiche Formulierung) wie bei den Prüfungen der letzten Jahre. Wenn man diesen Fragenkatalog also durchgearbeitet hat sollte der Theorieteil der Prüfung kein Problem mehr darstellen.

Alle Fragen, die bei der Prüfung am 21.01.2013 gekommen sind, sind bereits in frühreren Prüfungen vorgekommen. Diese Fragen, und jene, die in den alten Prüfungen öfter als 1x vorkommen (worauf anscheinend Wert gelegt wird) ist mit (!!!) markiert. **Viel Spaß!**

Software Wartung (!!!)

- Vervollständigen Sie die Tabelle um die **Wartungskategorien**, die in der Vorlesung besprochen wurden und geben Sie jeweils ein konkretes Beispiel an.
- Welche **Phasen** umfasst der **Wartungsprozess**?
- Nennen und beschreiben Sie zwei **Techniken zur Wartung** von Software.

Software Prozesse: Agile Softwareentwicklung (!!!)

- Erläutern Sie anhand einer Skizze das Grundkonzept von **SCRUM Sprints**.
- Erklären Sie den Unterschied zwischen dem **Produkt-Backlog** und **Sprint-Backlog**.
- Was versteht man unter einem **Burndown-Chart**?

Traceability (!!!)

- Was verstehen Sie unter **Traceability** im Zusammenhang mit Softwareprojekten?
- Beschreiben Sie kurz die unterschiedlichen **Arten von Traceability** und welche **Vorteile** sich **in einem Softwareprojekt** daraus ergeben.
- Wie kann **Traceability** in einem Softwareprojekt **umgesetzt** werden?

Qualitätssicherung und Testen (!!!)

- Diskutieren Sie die Begriffe **Verifikation** und **Validierung**.
- Erläutern Sie die folgenden Test-Design-Techniken: **Äquivalenzklassenzerlegung** und **Grenzwertanalyse**.
- Aus welchen **Komponenten** besteht eine **Testfalldokumentation**? Beschreiben Sie die notwendigen Komponenten und geben Sie ein **konkretes Beispiel** an. Hinweis: Verwenden Sie eine tabellarische Darstellung zur besseren Strukturierung der Testfallbeschreibung.
- Skizzieren sie **Black Box**- und **White Box-Testing**.
- Welche **Testarten** gibt es?
- Welche **Qualitätskriterien** gibt es in **der Softwareentwicklung**? Beschreibungen Sie diese.

Test-Driven Development (!!!)

- Was versteht man unter **Test-Driven Development** (TDD)? Beschreiben und skizzieren Sie die grundlegende Idee von TDD.
- Welche **Vorteile** ergeben sich aus der **Integration von TDD** im Rahmen der **Continuous Integration**?

Software Reviews (!!!)

- Was sind Software **Reviews** und wozu werden sie typischerweise eingesetzt?
- Welche **Arten von Reviews** kennen Sie? Erläutern Sie die wesentlichen Aspekte der einzelnen Reviewarten und geben Sie jeweils ein **konkretes Beispiel** an.
- Welche **Rollen** finden sich typischerweise **bei Reviews** und welche **Aufgaben** nehmen die unterschiedlichen Rollen wahr.

Integration (!!!)

- Erläutern Sie den Begriff der **Systemintegration**.
- Skizzieren und erläutern Sie die vorgestellten **Integrationsstrategien** und welche **Vor**- bzw. **Nachteile** damit verbunden sind.
- Was ist bei der **Systemintegration** im Hinblick auf die **Qualitätssicherung** zu beachten?
- Welche **Integrationsarten** exisiteren bzw. was ist Integration überhaupt?

Architektur (!!!)

- Beschreiben Sie das **Data Access Object Pattern** (DAO) anhand eines UML-Diagramms und erläutern Sie dessen Funktionsweise. Wann kommt dieses Pattern typischerweise zum Einsatz?
- Beschreiben Sie das **Factory Pattern**. Wann wird es eingesetzt? Geben Sie ein einfaches Beispiel in UML an.
- Welche **3 Typen von Pattern** gibt es? Nennen sie zu jedem mindestens 2 Beispiele.

Projektmanagement – Strukturpläne

- Was verstehen Sie unter einem **Projektstrukturplan**? (!!!)
- Aus welchen **Ebenen** besteht ein **Projektstrukturplan**? (!!!)
- (Skizzieren und erläutern Sie einen (i**) funktions- bzw. aufgabenorientierten Projektstrukturplan** und (ii) einen **objektorientierten Projektstrukturplan**. Worin liegen die wesentlichen **Unterschiede** und wofür werden sie eingesetzt?)

Projektmanagement

- Was ist ein **Projekt** und durch welche **Merkmale** ist ein Projekt gekennzeichnet? (!!!)
- (Was versteht man unter einem **Projekterfolg**?)
- Durch welche **Rahmenbedingungen** wird der **Projekterfolg gefördert**? Geben Sie für jede Rahmenbedingung zumindest ein konkretes Beispiel an.
- Was versteht man unter dem **Projektauftrag**? (!!!)
- (Was sind die **Aufgaben** eines **Teamkoordinators**?)
- Was ist **Projektmanagement**?
- Projektmanagement - **Erfolgsfaktoren**

Anpassung von Software-Prozessen (!!!)

- Erklären Sie die wesentlichen Konzepte des „**Process Customizing**“ und des „**Prozess Tailoring**“. Wozu und wann werden sie eingesetzt?

Requirements (!!!)

- Welche **Rolle** spielen **Requirements** im SE Life-Cycle?
- Wie können sie **klassifiziert** werden?

Design-Prinzipien (!!!)

- Beschreiben Sie das Design-Prinzip "**System Control**".
- Stair vs. Fork (+ Beispiele)
- **4+1 View Model Architecture**

Vorgehensmodelle

- **V-Modell** Prinzip + Vorteile + welche Zweige/Ebenen gibt es (!!!)
- **V-Modell XT** + Tailoring + Einsatzgebiet
- (**RUP Prinzip** + Vorteile + Phasen + Einsatzgebiet)

Sonstiges

- Was ist **Continuous Integration**?
- (**4 Kernformen** der Arbeitsstruktur)
- Wie sieht eine moderene **3-Tier-Architektur** aus? Beschreiben Sie die folgenden Fälle: a) **Thin-Client**, b) **Fat-Client**. Beschreiben und skizzieren Sie die beiden Architekturvarianten.