Testate 5. Vorlesung

Dienstag, 7. März 2017

15:52

**1. Versionsmanagement**

**Lernziele:**

* Sie sollen am Beispiel einen Änderungsmanagement-Prozess kennen

und erläutern können

**Definition:**

Änderungsmanagement: dieser Aufgabenbereich beschäftigt sich damit, die Änderungsanträge ( changerequests) der Kunden und Entwickler für die Software im Auge zu behalten, die Kosten und Auswirkungen dieser Änderungen zu ermitteln und zu entscheiden, ob und wann diese Änderungen implementiert und ausgeliefert werden sollen.

**Beispiel:**

Dies kann zum Beispiel ein in sich abgeschlossenes BackLog Item (Aus dem Agile Umfeld zur Beschreibung welcher in einem Block/Springt abgearbeitet werden kann) sein, durch das die Funktionalität erweitert wird.

Dies kann auch eine Korrektur sein, durch die ein Fehlverhalten der Software beseitigt wird.

* Sie sollen den typischen Versionsmanagementprozess kennen und

erläutern können

* Dient der Verwaltung und Verfolgung von verschiedenen Versionen von

Software

* Ermöglicht zeitgleiche Bearbeitung von Sourcefiles und ermöglicht das

Abmischen von Änderungen

Zur Unterstützung einer unabhängige Entwicklung ohne Konflikte verwenden

Versionsmanagementsysteme das Konzept eines öffentlichen Datenarchivs

(repository) und eines privaten Arbeitsbereichs.

Computergenerierter Alternativtext:
Arbeitsbereich (Ul) 
Arbeitsbereich (U2) 
x 
Ariane 
check in 
check out 
check In 
Wolfgang 
check out 
Versionsmanagementsystem 
Abbildung 25.8: Ein- und Auschecken aus einem Versionsarchiv. 

* Sie sollen GitHub als ein bekanntes Versionsmanagementsystem

benutzen können

erledigt

**Kontrollfragen**:

* Können Sie die Begriffe Versionsmanagementsystem und Version erklären?

**Versionsmanagement:**

Hierbei werden mehrere Versionen von Systemkomponenten verwaltet und es wird sichergestelt,

dass die Änderungen verschiedener Entwickler an den Komponenten sich nicht gegenseitig beeinträchtigen.

(Ian Sommerville)

**Version:**

Eine Instanz eines Konfigurationselements, die in irgendeiner Form von anderen Instanzen des Elements abweicht. Versionen haben immer einen eindeutigen Bezeichner, der sich oft aus dem Namen des Konfigurationselements und einer Versionsnummer zusammensetzt.

* Können sie den Unterschied zwischen Änderungsmanagement und Versionsmanagement erklären?

Siehe oben

* Können sie 3 verschiedene Versionsmanagementsysteme nennen?

CVS, Git, Subversion

**2. UI Design Prozess**

**Lernziele und Kontrollfragen**

* Sie sollen den Begriff Benutzbarkeit erklären können, sowie Merkmale und

Prinzipen der Benutzbarkeit nennen können.

**Definition:**

Fähigkeit des Softwareprodukts, vom Benutzer verstanden und benutzt zu werden sowie für den Benutzer erlernbar und ‚attraktiv‘ zu sein, wenn es unter den festgelegten Bedingungen benutzt wird.

**Merkmale:**

* Verständlichkeit
* Erlernbarkeit
* Bedienbarkeit
* Attraktivität
* Konformität

**Prinzipien:**

* Prinzip der Verbalisierung
* Architekturprinzip Ökonomie
* Architekturprinzip Trennung von Zuständigkeiten

Computergenerierter Alternativtext:
Prinzip 
Benutzervertrautheit 
Konsistenz 
Minimale Überraschung 
Wiederherstellbarkeit 
Benutzerführung 
Benutzervielfalt 
Beschreibung 
Die Oberfläche sollte Bezeichnungen und Begriffe aus der Erfahrungswelt 
der Menschen verwenden, die am meisten vom System Gebrauch machen. 
Die Oberfläche sollte, wo immer möglich, konsistent sein. Das bedeutet, 
dass vergleichbare Operationen auf dieselbe Weise veranlasst werden 
sollten. 
Benutzer sollten vom Systemverhalten nie überrascht werden. 
Die Oberfläche sollte Wiederherstellungsmechanismen für den Benutzer 
enthalten. 
Die Oberfläche sollte beim Auftreten von Fehlern aussagekräftige Rück- 
meldungen und kontextsensitive Hilfsmittel für Benutzer anbieten. 
Die Oberfläche sollte für verschiedene Arten von Systembenutzern 
geeignete Interaktionsmöglichkeiten bereitstellen. 
Abbildung 16.1: Grundsätze für den Entwurf von Bedienoberflächen 

* Sie sollen den Begriff Ergonomie erklären und abgrenzen können sowie die

Grundsätze zu Dialoggestaltung nennen können.

**Definition:**

Die Ergonomie ist die wissenschaftliche Disziplin und systematische Forschung, die sich mit dem Verständnis der Wechselwirkungen zwischen menschlichen und anderen Elementen eines Systems befasst, und der Berufszweig, der Theorie, Prinzipien, Daten und Methoden zur Gestaltung von Arbeitssystemen anwendet, um das Wohlbefinden des Menschen und die Leistung des Gesamtsystems zu optimieren.

**Grundsätze der Dialoggestaltung**

* Aufgabenangemessenheit
* Selbstbeschreibungsfähigkeit
* Erwartungskonformität
* Lernförderlichkeit
* Steuerbarkeit
* Fehlertoleranz
* Individualisierbarkeit

* Sie können die einzelnen Schritte in einem interaktiven UI-Entwicklungsprozess beschreiben?

Der Prozess des Oberflächenentwurfs ist ein iterativer Vorgang. Erste Entwürfe sollten auf Papier entwickelt und getestet werden.

Der Gesamtvorgang des Oberflächenentwurfs besteht aus 3 Kerntätigkeiten:

* Benutzeranalyse:

Bei der Benutzeranalyse eignen Sie sich Kenntnisse über die Aufgaben, die die Benutzer erfüllen müssen und über ihr Arbeitsumfeld an.

* Erstellen des Systemprototypen:

Zwar können sich die Benutzer über die Möglichkeiten äußern, die sie in einer Oberfläche benötigen, doch ist es sehr schwer für sie, genaue Angaben zu machen, solange sie nichts Konkretes vor Augen haben. Daher müssen Sie Systemprototypen entwickeln und den Benutzern vorstellen, die dann die Weiterentwicklung der Oberfläche steuern können.

* Oberflächenbewertung:

Sicherlich sprechen Sie während der Entwicklung der Prototypen mit den Benutzern, doch sollten Sie auch eine formalere Bewertung wählen, wenn Sie Informationen über die Erfahrungen der Benutzer mit der Oberfläche sammeln.

* Können Sie Prinzipien von User CenteredDesign (UCD) nennen?

Computergenerierter Alternativtext:
In a Nutshell, the 4 Principles of UCD 
Focus on real end-users 
Validate requirements & designs 
Design, prototype, & develop products iteratively 
Understand & design for a holistic user experience 

**3. UI Framework**

**Lernziele**

* Sie sollen das klassische MVC Pattern erläutern können

Computergenerierter Alternativtext:
Model-View-ControIler Muster 
Controller (Steuerung) 
verbindet Benutzeraktionen 
mit Aktualisierungen 
des Modells wählt die 
Präsentation aus 
Zustands- 
änderung 
Auswahl der 
Präsentation 
Benutzer- 
ereignisse 
View (Präsentation) 
stellt Modell dar 
fordert Aktualisierung des 
Modells an 
sendet Benutzerereignisse 
an die Steuerung 
Benachrichtigung 
über Änderung 
Zustands- 
abfrage 
Model (Modell) 
kapselt Anwendungszustand 
meldet Zustandsänderung 
an die Präsentation 
Abbildung 6.3: Die Struktur von MVC. 

Computergenerierter Alternativtext:
Name 
Beschreibung 
Beispiel 
Verwendung 
Vorteile 
Nachteile 
MVC (Model-View-ControIIer) 
Trennt Präsentation und Interaktion von den Systemdaten. Das System ist in drei logi- 
sche Komponenten unterteilt, die miteinander interagieren. Die „Model "-Kompo- 
nente verwaltet die Systemdaten und die mit diesen Daten verbundenen 
Operationen. Die „View"-Komponente definiert und verwaltet die Art und Weise, wie 
die Daten dem Benutzer zur Verfügung gestellt werden. Die „Controller "-Kompo- 
nente verwaltet die Benutzerinteraktion (z. B. Tastendruck, Mausklick usw.) und 
reicht diese Interaktionen an „View" und „Model" weiter. Siehe Abbildung 6.3. 
Abbildung 6.4 zeigt die Architektur eines webbasierten Anwendungssystems, das 
mithilfe des MVC-Musters organisiert ist. 
Wird eingesetzt, wenn es verschiedene Sichten auf die Daten und Möglichkeiten der 
Interaktion mit diesen gibt. Wird außerdem verwendet, wenn die zukünftigen Anfor- 
derungen an Interaktion und Präsentation der Daten noch nicht feststehen. 
Ermöglicht es, dass die Daten unabhängig von ihrer Repräsentation geändert werden 
können und umgekehrt. Unterstützt die Darstellung derselben Daten auf verschie- 
dene Arten, wobei Änderungen, die in einer der Darstellungen vorgenommen wer- 
den, in allen sichtbar sind. 
Kann zusätzlichen Code und Codekomplexität mit sich bringen, wenn das Daten- 
modell und die Interaktionen einfach sind. 
Abbildung 6.2: Model-View-Controller (MVC). (Forts.) 

* Sie sollen die neueren Varianten des MVC Ansatzes benennen und erklären können

Computergenerierter Alternativtext:
View 
Model 

**Model-View-Presenter**

* Klare Entkopplung von View und Model
* Der Presenterübernimmt die Controlleraufgaben
* Aktualisiert den View bei Modell-Änderungen

Computergenerierter Alternativtext:
Model-View-ViewmodeI 
Command-Binding 
View 
Data-Binding 
ViewModel 
Model 

* Sie sollen das MVC Pattern oder seine Varianten in ihrer Anwendung umsetzen können

Erledigt

**Kontrollfragen:**

* Können sie die Unterschiede von MVC und MVVM beschreiben?

In der GUI-Entwicklung wird MVC in der Regel wie folgt verwendet:

* View wird angezeigt, liest dafür Daten aus dem Model aus.
* Interaktion wird vom View an den Controller weitergegeben.
* Controller ändert das Model und wählt einen View aus.

Bei MVVM sieht es anders aus:

* View und ViewModel interagieren miteinander. Der View übernimmt einen Teil der Controller-Tätigkeiten und reagiert auf Benutzerinteraktionen. Als Konsequenz dieser Interaktionen ändert er das ViewModel.
* Das ViewModel reagiert auf diese Änderungen und gibt sie an das Model weiter.
* Änderungen am Model müssen im ViewModel gespiegelt werden, damit sie im View angezeigt werden können.

Wo ist nun der Unterschied?

1. Bei MVC ist der View viel einfacher. Er nimmt keine Änderungen vor, sondern nutzt grundsätzlich den Controller (bzw. Presenter) für die Logik. Bei MVVM reagiert der View auch auf Änderungen am ViewModel (über z. Bsp. INotifyPropertyChanged) und ändert selbstständig die Darstellung. Der View ändert auch das ViewModel, wenn zum Beispiel Text in einem Eingabefeld geändert wird.
2. Ein Controller ist in der Regel stateless, er erhält sein Daten als Aufrufargumente. Das ViewModel enthält selbst Daten und ist dadurch an eine View-Instanz gebunden.
3. Ein ViewModel enthält auch die View-Logik. Idealerweise wird für die Views im MVVM kein Code-Behind benötigt, sondern alles über DataBinding, Actions und das ViewModel erledigt.

* Können sie an einem Codingbeispielden MVVM-Ansatz von AngularJS erläutern

Siehe Folie 20

* Können Sie den Unterschied zwischen einer klassischen und einer Modernen Webapplikation erklären?

**Die moderne Web-Anwendung:**

Bei SPA Anwendungen, Moderne Web-Anwendung, spielt sich ein Großteil der Logik auf Clientseite ab, somit ist vieles in JavaScript implementiert.

* Grundlage war die Einführung des XMLHttpRequest-Objekt mit Internet Explorer 5
* programmatisch ( auch asyncron) vorbearbeitete HTML Schnipsel werden vom Server geladen und dynamisch in das DOM einbauen.
* Später hat man dann mehr und mehr nur auf das Model zugegriffen und den View komplett auf Clientseite aufgebaut.
* Ursprünglich ging der Datenaustausch über XML ( AJAX = asyncronousJavaScript andXML). Heute wird vor allem JSON als Format verwendet.

**4. REST Services**

* Können sie 2 Prinzipien von RESTful Services benennen und erklären?
  + Unterstützung der Client –Server Architektur
  + Stateless ( -> Skalierbarkeit )

Es werden Daten über HTTP übertragen , ohne eine

Sitzungsverwaltung über Cookies einzusetzen

* Caching ( -> Performance)
* Einheitliche Schnittstelle:
  1. Adressierbarkeit von Ressourcen (URI)

Jede Ressource muss über eine Route verfügbar sein. z.B. <https://myapp.com/rest/restaurant>

1. Repräsentationen zur Veränderung von Ressourcen (JSON, XML, etc)

Die Ressource der Datenbank wird z.B. Als JSON geliefert und nimmt auch JSON zum speichern wieder an.

1. Selbstbeschreibende Nachrichten ( HTTP Methoden)

Es wird mittels über die gewählte Http-Methode entschieden, was getan wird.

**Hinweis:**

GET – Frage die Repräsentation einer Ressource ab. GET Anfragen können

beliebig oft gestellt werden und dürfen keine Nebeneffekte haben

POST – Einer Ressource kann etwas hinzugefügt werden . POST –Anfragen

können Seiteneffekte haben

PUT – Mit PUT können neue Ressourcen erzeugt werden oder der Inhalt

bestehender Ressourcen ersetzt werden

DELETE – Ressourcen können gelöscht werden

* Können sie den Begriffe oDATA und den Zusammenhang mit RESTful Service erklären?

Ein standardisiertes Protokoll auf Basis von HTTP und REST ist und die CRUD Services (Create, Read, Update, Delete) unterstützt, um Operationen auf Daten zu ermöglichen (data APIs).

* Können Sie das Datenmodell von oData erläutern?

Das Datenmodell auf dem Server besteht aus ( Link ):

* Entity sets von Entity types
* Eigenschaften / Properties
* Associationen zwischen Entity sets
* Navigationen

**Hinweis:**

Computergenerierter Alternativtext:
http://<host>/<ServiceRoot>/<Products 
Enuy_Seg: EDM basic 
building block; 
instances Of entity types 
Nayjga.llQn: Implements 
an association 
Properties 
ausinessPartnerID 
CompanyName 
Country 
WebA 
City 
postalCode 
Navigation Properties 
SalesOrder5et 
Sets 
Properties 
salesOrdedD 
• - ü BuyerName 
BuyerID 
Navigation Properties 
Business P 
ASSQCjatiQn: Relationship 
between two entity types 
EDM basic 
building block; 
represents specific 
business Object 
At least one property 
must be defined as a 
Isey field. 
An entity type is built 
from one or more 
Properties. 