Praktikum 4 - Architekturbewertung

Praktikum Architektur von Informationssystemen

Sommersemester 2016 – Aufgabe 4

Nils Löwe , Tel. 0176 / 574 550 65

Rahmenbedingungen für das Praktikum

- Die Bearbeitung findet in 3er-Gruppen statt.
- Die Präsentation Ihrer Lösungen erfolgt im Rahmen der Praktikumstermine.
- Jede Gruppe hat dazu 15-20 Minuten Zeit. Bereiten Sie hierzu Folien vor.

ACHTUNG: Alle Gruppen präsentieren Ihre Lösung in der letzten Vorlesungswoche am 9.6.!

Aufgabe 4: Architekturbewertung

Hintergrund: Neben dem Architekturentwurf den Sie für das letzte Praktikum erstellt haben, wurde von einem Nachbarteam ein weiterer Entwurf vorgeschlagen. Sie sollen den vorgeschlagenen Entwurf bewerten und die Vor-/Nachteile gegenüber Ihrem eigenen Entwurf analysieren und präsentieren.

Der Rahmen des Projekts ist die zuvor beschriebene Zeitplanungssoftware für einen Anbieter von Wartungen von Windenergieanlagen. Die Software hat eine Anbindung die Kundendatenbank und die Mitarbeiterdatenbank. Als Kundendatenbank wird "SugarCRM" eingesetzt und die Mitarbeiterdatenbank ist eine einfache MySQL Datenbank.

Das Ergebnis dieser Aufgabe soll wieder eine reine Präsentation sein, es muss kein Code geschrieben werden

Aufgabenbeschreibung

Bewerten Sie den unten vorgeschlagenen Architekturentwurf in Bezug auf die folgenden Kriterien:

- Vollständigkeit des Entwurfs (inhaltliche Anforderungen)
- Können die nichtfunktionalen Anforderungen erfüllt werden?
- Eignung der vorgeschlagenen Technologien
- Komplexität des Entwurfs
- Umsetzbarkeit in mehreren Teams

Vergleichen Sie anschließend den vorgeschlagenen Architekturentwurf mt Ihrem eigenen aus dem letzten Praktikum und stellen Sie die Unterschiede heraus. Wo hat der vorgeschlagene Entwurf Stärken und wo hat er Schwächen gegenüber Ihrem eigenen Entwurf?

Aufgabe der geplanten Softwrae

Use Cases

- Eine Servicekraft greift lesend auf seinen/ihren Wochenplan zu.
- Ein "Planer" erstellt 1x/Woche die Wochenpläne.
- Servicekräfte werden per E-Mail über neue Pläne, bzw. Änderungen informiert.

Funktion: Zugriff auf die Pläne

- Die Pläne sollen über ein Web-Interface mit user/Passwort login lesbar sein.
- Die Pläne sollen zusätzlich über eine REST API abrufbar sein, um ggf. Android/iPhone Apps zu unterstützen.

Funktion: Erstellen der Pläne

- Die Pläne sollen teil-automatisiert erstellt werden. Basierend auf den Vorgaben macht die Software einen ersten Vorschlag, der dann manuell geprüft und vervollständigt wird.
- Die Pläne werden an einem Bürorechner erstellt, ob die Software eine Webanwendung oder eine Desktopanwendung ist, ist dabei egal.
- Die Vorgaben für die Pläne bestehen aus der Verfügbarkeit der Servicekräfte, den anstehenden Wartungsterminen für die Woche und dem prozentualen Puffer, der für jede Servicekraft für Notfälle vorgehalten werden soll.
- Die Verfügbarkeit und der Puffer werden direkt aus der MySQL Datenbank gelesen.
- Die Wartungstermine werden per REST API aus dem SugarCRM ausgelesen. Ein Wartungstermin enthält u.a. die Angaben über Anfahrtdauer und Dauer der Wartung (1 Stunde min / 2 Tage max.)
- Ist ein Plan fertig, soll das System eine E-Mail an die Servicekraft versenden.

Nutzergruppen

- Etwa 150 User greifen als Wartungspersonal auf ihre Wochenpläne zu. Es erfolgt ein einfacher, lesender Zugriff
- 1-3 Planer erstellen die Pläne 1x/Woche
- Bei den Servicekräften wird mit einem jährlichen Wachstum von 10-20 Personen gerechnet

Architekturentwurf

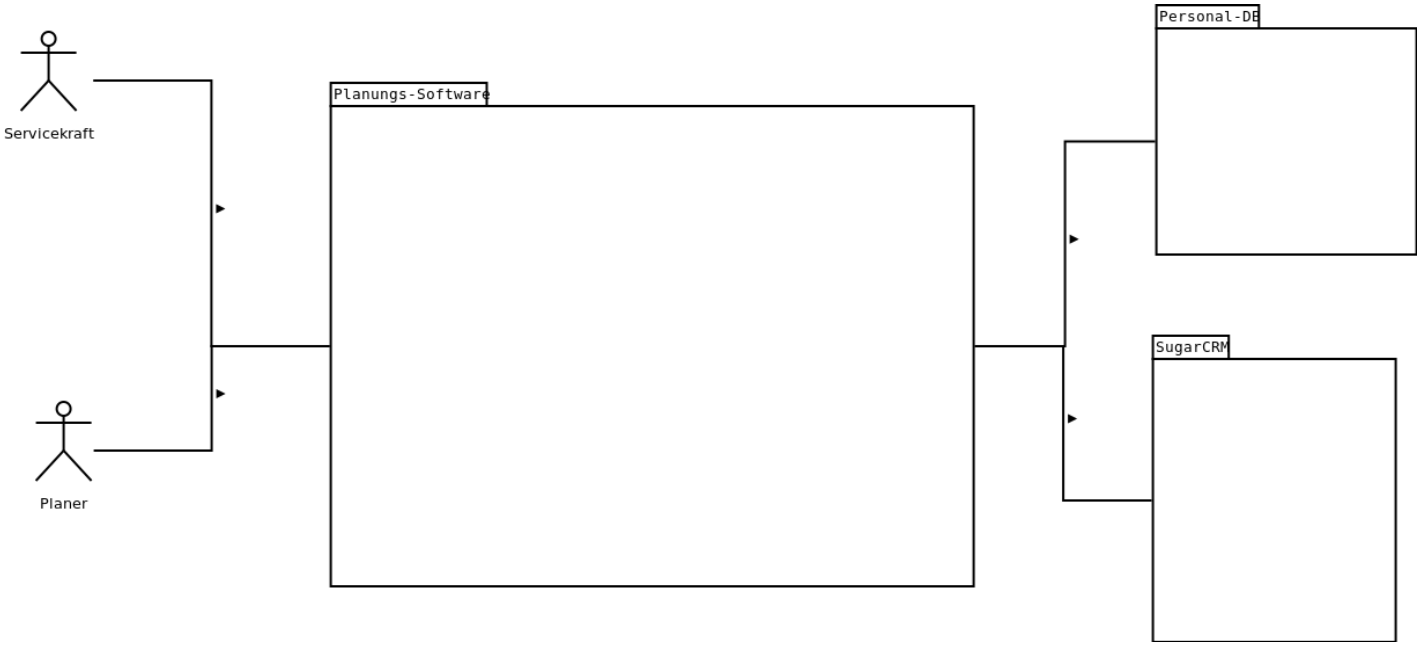
Überblick

Die geforderte Software wird als Microservice Architektur entwickelt. Um die geforderten Funktionen zu gewährleisten, wird die folgende Gruppe von Services entwickelt:

- User-Service: Synchronisation mit der Personal-DB über AMQP, Login, Authentication

- Customer-Service: Synchronisation mit dem SugarCRM über AMQP
- Plan-UI: Web-App zum Erstellen (Plan-Service) und Lesen der Pläne (Plan-DB)
- Plan-Service: Teilautomatisierte Erstellung der Pläne mittels Blackboard Agenten
- Plan-DB: Speicherung der erstellten Pläne
- Email-Service: Senden von Emails an Servicekräfte
- External API: Zugriff für mobile Apps

Kontextsicht



Eingesetzte Technologien

User-Service

Der User-Service wird als NodeJS App in Javascript implementiert. Eine interne API bietet Login/Logout und Userdaten an. Mittels AMQP wird jede Nacht die Datenbank des User-Service mit der MySQL Personaldatenbank synchronosiert.

Customer-Service

Der Customer-Service wird als Phoenix App in Elixir implementiert. Der Service stellt im Wesentlichen ein Abbild der benötigten Daten aus dem SugarCRM zur Verfügung, die damit innerhalb der Planungssoftware gespiegelt werden. Mittels AMQP wird jede Nacht die Datenbank des Customer-Service mit dem SugarCRM synchronosiert.

Plan-UI

Die Plan-UI wird als RubyOnRails App implementiert. Als Frontend wird Twitter Bootstrap zusammen mit jQuery eingesetzt.

Plan-Service

Der Planungs-Service wird als NodeJS App in Javascript implementiert.

Plan-DB

Die Plan-DB wird als PostgreSQL Datenbank umgesetzt, da sich diese einfach als Master/Slave Konfiguration spiegeln lässt. Um die Plan-DB einfach zugreifbar zu machen, wird eine RubyOnRails App um die Datenbank herum gebaut.

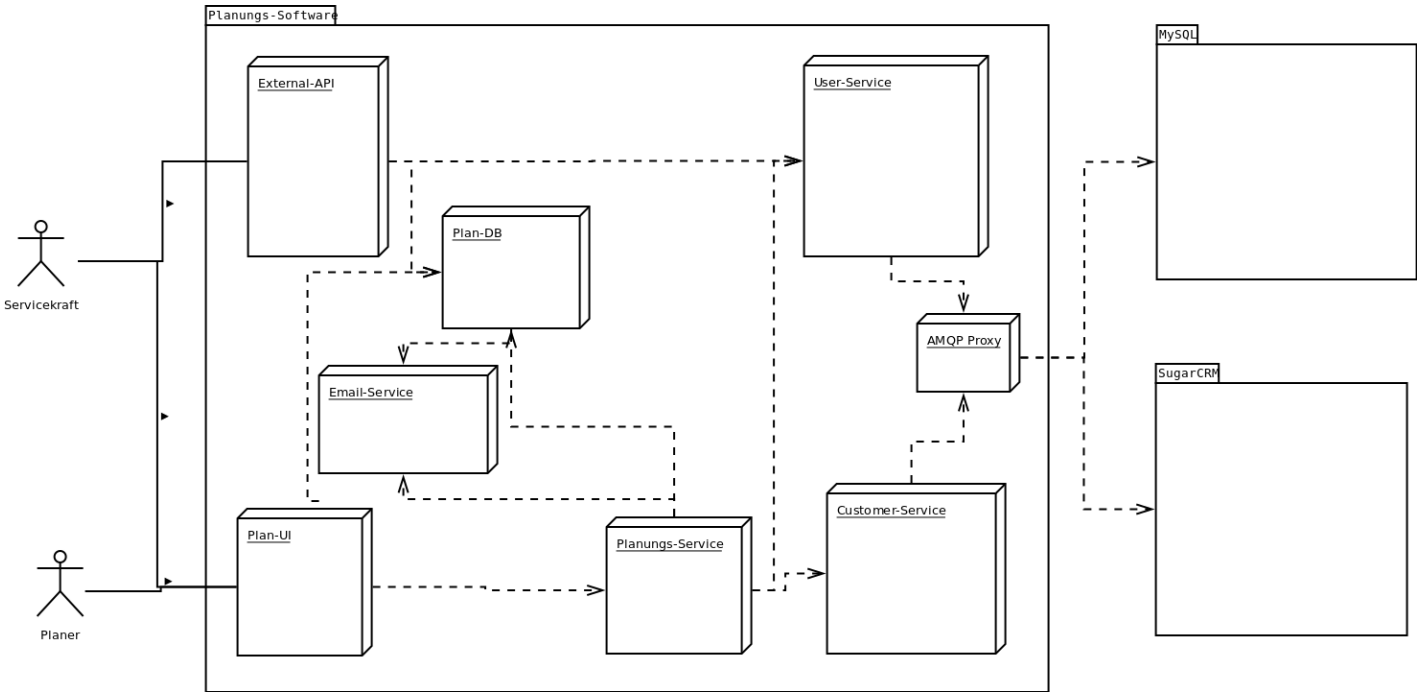
Email-Service

Der Email-Service wird als NodeJS Service in Javascrip implementiert.

External API

Die External API wird als mit Yesod in Haskell implementiert. Die API bietet lediglich einen Lesezugriff auf die Plan-DB und nutzt zur Authentication den User-Service.

Verteilungssicht



Verfügbarkeit

Um die Verfügbarkeit von 99,9% zu sichern, wird folgendes Setup vorgeschlagen

- Einsatz von Docker / Docker Swarm als Deployment Plattform
- Betrieb von drei unabhängigen "Swarms" aus jeweils vier Servern in drei Zonen
- Einsatz von "Consul" als Discovery und Management Service
- Datenbanken als Master/Slave über alls drei Zonen gespiegelt
- Als Hoster wird Amazon AWS vorgeschlagen

Sicherheit

Nach außen ist nur die External API erreichbar. Während der Entwicklung wird ein besonderer Fokus in der Code Reviewws auf die Sicherheit der API gelegt. Die Plan-UI ist nur aus dem Intranet erreichbar.

Alle Verbindungen zwischen den Services werden mit SSL verschlüsselt.

Für die API wird oAuth2 aus Sicherung eingesetzt.

Ihre Aufgabe

Analysieren Sie den vorgeschlagenen Entwurf in Bezug auf Vor-/Nachteile und auf die Eignung zur Erfüllung der funktionalen und nichtfunktionalen Anforderungen der vergangenen Aufgabenstellung.

Erstellen Sie für den Praktikumstermin eine Präsentation in der Sie die Ergebnisse Ihrer Analyse vorstellen. Beantworten Sie dabei die folgenden Fragen:

- Erfüllt der Entwurf alle funktionalen Anforderungen?
- Erfüllt der Entwurf alle nichtfunktionalen Anforderungen?
- Erscheint Ihnen der Entwurf angemessen komplex? Ist er zu kompliziert und vereinfacht er zu stark?
- Lässt sich die Software ggf. in mehreren parallelen Teams entwickeln, oder ist die Kopplung der Komponenten zu eng?
- Wo hat der Entwurf Vor- und Nachteile gegenüber Ihrem Entwurf aus der letzten Aufgabe?

Die Präsentation ist bis zum Praktikumstermin per Mail an nils@loewe.io abzugeben. In der Praktikumszeit sind keine Änderungen mehr vorgesehen.