1. PHASE

KONSTITUIERUNG DES TEAMS UND ANALYSE DER EIGENEN STÄRKEN UND SCHWÄCHEN

Risikoanalyse

abs16
12. Dezember 2015

EINLEITUNG

Durch unser bisher begrenztes Wissen bezüglich (Multi)-Agentensystemen fällt uns ein Einschätzen der Risiken verhältnismäßig schwer. Da unser Team aus sehr unterschiedlichen Charakteren mit diversen Fähigkeiten besteht, sind wir davon überzeugt, für jedes aufkommende Problem eine Lösung zu finden.
Wir kennen uns bereits seit Beginn des Studiums und haben in arbeitsintensiven Phasen zusammengearbeitet. Daher dürfte die interne Kommunikation kein Problem darstellen. Desweiteren sind wichtige Gebiete wie das Java-Programmieren, Projekt-Management, Qualitätssicherung und Infrastruktur schon jetzt durch vorrangehende Interessen und Erfahrungen einzelner gut abgedeckt.
Das ganze Team ist hoch motiviert ein herausforderndes Projekt zu bearbeiten.

HOHE SELBSTÜBERSCHÄTZUNG

Die größte Gefahr, welche unserer Gruppe droht, ist, dass die guten Voraussetzungen, welche wir mitbringen, dafür sorgen, dass wir unsere Leistungsfähigkeit überschätzen. Wir glauben, durch ständiges Reviewing dafür sorgen zu können, uns gegenseitig “auf den Boden der Tatsachen” zurück zu bringen.

FOKUS VERLIEREN

Durch die Größe des Projektes aber auch die Selbsteinschätzung von uns selbst kann es dazu kommen, dass wir das eigentliche Ziel des Projektes aus den Augen verlieren und uns zu sehr

auf zusätzliche Features statt auf die Fertigstellung des eigentlichen Projektes konzentrieren. Um dieses Problem zu adressieren, ist es wichtig, uns stetig bei unseren Treffen abzustimmen und regelmäßige Einschätzungen unserer Betreuer zu nutzen.

Aufgaben falsch verstehen

Durch unsere bereits angesprochene Unerfahrenheit kann es dazu kommen, dass wir die an uns gestellten Anforderungen falsch verstehen. Eine umfassende Einarbeitung mithilfe von gegebener Literatur und eine genaue Analyse der Aufgaben sollten genügen um diese Gefahr abzuwenden.

Aufwand falsch einschätzen

Da wir noch nie gemeinsam an einem so komplexen Projekt gearbeitet haben, kann es sein, dass wir den Aufwand - vor allem den des Programmierens - falsch einschätzen. Verhindern können wir diese Fehleinschätzung durch Absprache mit unseren Betreuern und das Hinzuziehen ihrer Einschätzung. Ansonsten versuchen wir uns an dem vorgegebenen 150 Stunden Workload für das Modul zu orientieren.

Schlechtes Zeitmanagement

Weil sich die Projektgruppe täglich sieht, können wir verhindern, dass Aufgaben aufgeschoben werden. Weiterhin besteht bei allen Gruppenmitgliedern das Interesse, den Großteil der Programmierung in den Semesterferien zu bewältigen, so dass eine Zeiteinteilung sehr einfach ist.

Ausfall eines oder mehrerer Gruppenmitglieder

Wir möchten durch unsere wöchentlichen Meetings für eine möglichst symmetrische Informationsverteilung sorgen, so dass Rollen schnell die ausführende Person wechseln könnten, sollte ein Gruppenmitglied krankheitsbedingt (oder ähnliches) ausfallen.

Ausfall des Servers und Datenverlust

Wenn der Server, auf dem die gemeinsame Codebasis sowie andere Daten gehostet sind, ausfällt, darf dies nicht zu einer Behinderung für unserer Arbeit werden. Durch die Nutzung von Git steht jedoch jedem Teammitglied immer eine relativ aktuelle Version lokal zur Verfügung. Außerdem stehen uns alternativ eigene Server zur Verfügung, so dass falls ein Ausfall auftritt, dieser nur von sehr kurzer Dauer sein wird, bevor das gesamte Team weiterarbeiten kann. Um mögliche weitere Datenverluste zu kompensieren, werden wir regelmäßig ein Backup der Daten erstellen.

Fehlerhafte Projektanalyse am Anfang

Direkt zum Scheitern verurteilt ist das Projekt, wenn von Anfang an eine schlechte bzw. falsche Projektanalyse erfolgt. Durch sehr genaue Abstimmung mit den Betreuern und eine umfassende Recherche, versuchen wir das zu verhindern.

Technische Komplikationen

Da die Teammitglieder mit unterschiedlichen Systemen arbeiten und ganz persönliche Präferenzen haben kann es zu Komplikationen kommen, wenn es um den Umgang mit bestimmten Tools geht bzw. wenn eine konsistente Codebasis entstehen soll. Dieses Risiko versuchen wir einzudämmen, indem wir von Anfang an sehr genaue Spezifikationen festlegen und unsere Arbeit sehr genau dokumentieren.

Modellierungsfehler

Es kann passieren, dass unser modelliertes Produkt Inkonsistenzen aufweist oder Problematiken nicht lösen kann. Das lässt sich nur verhindern, indem wir eine umfassende Anforderungsanalyse schreiben und uns bewusst machen, welche Probleme wir lösen müssen.

Für Einige unbenutzbares Produkt

Da Menschen allgemein auf unterschiedlichen Systemen mit unterschiedlichen Voraussetzungen arbeiten, kann es passieren, dass unser Endprodukt nicht von jedem benutzbar ist. Durch die Analyse der Anforderungen können wir das Problem eingrenzen. Weiterhin können wir versuchen durch Webtechnologien die Bandbreite der erreichbaren Nutzer zu maximieren.