Virtuelle Jukebox

Mobile App Client – Resümee

David Böhm-Vrana
Mathias Dittrich
Tobias Egger
Paul Götzinger
Sophia Nunner

Projektnummer: PIE.XX.XXX
Version: 1.0
Datum: 18.12 2019

Änderungsverzeichnis

Versions	Datum	Änderung	Ersteller
1.0	18.12.2019	Erste Version	D. Böhm-Vrana, M. Dittrich, T. Egger, P. Götzinger, S. Nunner

Inhalt

1 Rückblick 4

2 Was lief schief und warum? 4

3 Was ging gut und warum? 5

4 Erfahrung mit Scrum? 6

1 Rückblick

Zwar ist die Android-Entwicklung kein typisches HSD Projekt, den Exkurs wurde allerdings vom Projektteam als sehr interessant aufgefasst. Unabhängig vom persönlichen Vorwissen wäre ein (verpflichtender) Git-Kurs oder zumindest eine Integration in den Übungsbetrieb von SDP/BSY/PSS/ISE vorteilhaft, da der Umgang mit Git während dem Projekt das ein oder andere Mal schwierig war.

Die Organisation wurde im Projektteam generell gut aufgefasst und jedes Teammitglied wusste immer was zu tun ist. Die flexible Zeiteinteilung mit "Soft"-Deadlines wurde als angenehm aufgefasst. Kommunikation mit den anderen Teams (Server etc) verlief über eigene Meetings in kleiner Runde mit den einzelnen Product Ownern und verlief konstruktiv und entspannt.

Die Erstellung der Schnittstellendokumenten war für die Kommunikation mit anderen Teams und für die Implementierung sehr wichtig und es hätte einiges an Mehraufwand bedeutet, wenn diese nicht zu beginn sauber ausgearbeitet worden wäre.

Allerdings wurde die Erstellung von einigen weiteren Dokumenten die nicht unbedingt notwendig gewesen wären eher als Last angesehen und haben bis zum Abschluss des Projekts keinen relevanten Einfluss auf das Projektergebnis oder dessen Ablauf. Konkret geht es hierbei um die Erstellung der Projektkosten und Erstellung der Risikoanylse inklusive Risk Map. Die Frustration über die Erstellung liegt hierbei größtenteils daran, dass bis zum Abschluss nicht klar wurde welchen Nutzen diese Dokumente erfüllen, da weder durch ein entsprechendes Risikobudget das Risiko vermindert werden konnte und die Kosten wurden lediglich nach dem geschätzten Stundenaufwand berechnet, welches ebenfalls unrealistisch erscheint.

Das Arbeiten im Team und die Zusammenarbeit sowie der Informationsaustausch mit den anderen Teams war bestimmt eine gute Übung für künftige Projektarbeiten.

Durch die Aufteilung der Tasks zum Projektbeginn in kleinere Subtasks sowie der Definition der einzelnen Interfaces sowohl intern als auch zum Serverteam verlief die Teamarbeit reibungslos ab und das Projekt konnte somit mit nur kleineren Hürden bewältigt werden.

2 Was lief schief und warum?

- Erst falscher Ansatz bei der Softwarearchitektur durch die Verwendung von mehreren Activities. Es wurde klar, dass die App durch die Verwendung von mehreren Activities nicht entsprechend (oder zumindest mit unserem derzeitigen Wissen) umgesetzt werden konnte. Es musste somit ein Wechsel von mehreren Activities hin zur Verwendung von Fragmenten umgestellt werden. Dieser Wechsel bedeutete zwar einen Mehraufwand aber dadurch, dass dieser Design Fail relativ früh erkannt wurde hielt sich dieser Mehraufwand glücklicherweise in Grenzen.
  Der Grund wieso es überhaupt zu diesem Design Fehler gekommen ist, dass bei der Planungsphase diesem Aspekt zu wenig Aufmerksamkeit gewidmet wurde. Dies liegt vor allem an der relativ knappen und ambitioniert gewählten Deadline. Dadurch

wurde zu früh mit dem Schreiben des Codes begonnen ohne die Planungsphase ordentlich abzuschließen.

- Nicht 100% reproduzierbare Abstürze der App. Erst nach langen Debug-Sessions konnte die Ursache eruiert werden. Der Absturz lag hierbei an der zyklischen Serverabfrage und der Verwendung der Fragmente. Wird zwischen den Screens gewechselt so wird das Fragment immer ausgetauscht und die Daten für den entsprechenden Screen dynamisch zu dieser Zeit generiert. Um nun neu erhaltene Daten im Screen zu aktualisieren muss das entsprechende Fragment von dem Hintergrundprozess, welcher das Pollen des Servers übernimmt, benachrichtigt werden. Wird allerdings nun der Screen gewechselt erfolgt ein entsprechender Austausch der Fragmente was einige Zeit in Anspruch nimmt. Hier kam es gelegentlich zu der Situation wo der Hintergrundprozess das entsprechende Fragment benachrichtigen wollte dieses aber noch nicht komplett aufgebaut war und somit noch nicht vorhanden. Dadurch kam es zu dem Zeitpunkt der Benachrichtigung zu einem Absturz der App. Gelöst wurde dies indem der Hintergrundprozess erst gestartet wird, wenn der Aufbau des Fragments abgeschlossen ist, da wir auf den Aufbauprozess des Fragments selbst keinen Einfluss nehmen konnten.

3 Was ging gut und warum?

- Ambitionierte Projektteams wodurch die Ausarbeitung der Projektübergreifenden Schnittstellen (REST) recht rasch erfolgt ist. Vorschläge zu Adaptionen die während der Implementierungsphase entstanden sind wurden entsprechend diskutiert und übernommen. Durch diese Definition war es möglich die Implementierung Clientseitig bereits durchzuführen ohne noch einen entsprechenden Testserver zur Verfügung zu haben. Beim ersten Client-Server-Test zeigte sich, dass es dadurch ebenfalls nur zu geringen Komplikationen kam und die Integration eigentlich reibungslos über die Bühne lief.
- Entspanntes Arbeitsklima aufgrund hilfsbereiter und motivierter Kollegen sowohl im eigenen Team als auch Teamübergreifend. Informationsaustausch zwischen den einzelnen Teams erfolgte über die Product Owner um Overhead zu vermeiden, was sehr gut funktioniert hat. Ebenfalls die Aufteilung der einzelnen Tasks projektintern funktionierte reibungslos.
- Gemeinsames Debuggen in der Übung führte dazu, dass die Fehler schneller gefunden werden konnte wodurch sich ein effizienterer Ablauf bei der Implementierung ergab. Weiter wurden die Übungen dazu genutzt um sich über den Verlauf der vergangenen Woche auszutauschen und die Aufgaben für die kommende Woche zu spezifizieren.

4 Erfahrung mit Scrum?

- Bei dieser Projektgröße wurde dies eher als ein Overkill angesehen. Gerade wenn ein Subtask lediglich von einer Person abgearbeitet wird, was zu einem erhöhten bürokratischen Aufwand für diese „simplen“ Tasks geführt hat. Weiter ist es unrealistisch ein Daily Stand-Up abzuhalten, wodurch dieses entsprechend nicht abgehalten wurde.
- Deshalb auch Vernachlässigung von Trello. Burndowncharts, etc. da diese eher als Überflüssig für das umgesetzte Projekt angesehen wurden.
- Vgl. Aufwand - Nutzen von Scrum mit Erstellung der Tasks in Trello, Punktezuweisung, Personenzuweisung, Sprint Planning mit Verschieben der Tasks, die Boards auf dem Laufenden halten, Retrospective, etc. wurde nicht gut aufgenommen, da der Aufwand als zu hoch im Vergleich zum erhaltenen Nutzen angesehen wurde. Hinzu kam hierzu, dass die interne Projektsynchronisation zum Großteil über eine entsprechende Whatsapp Gruppe vorgenommen wurde wodurch der Vorteil von den Trello Boards weiter ins Hintertreffen gelangte.
- Auch die Rollenverteilung (Scrum Master, Product Owner) unrealistisch und unsinnig für das umgesetzte Projekt. Alle sind Entwickler und manche zusätzlich SM und PO - so sollte das in echten Projekten nicht sein.
- Da der Projektumfang doch eher überschaubar war wurde Scrum eher als Last anstelle einer Bereicherung angesehen. Womöglich wäre es besser bei dieser Projektgröße auf Canban Boards zu wechseln. Hierbei erfolgt noch immer eine Aufteilung in einzelne Tasks, was durchaus positiv aufgenommen wurde, aber jeder Entwickler kann selbst einen Subtasks für sich beanspruchen, wenn dieser nicht ausgelastet ist. Weiter kann dadurch trotzdem noch transparent dargestellt werden wer gerade an welchem Task arbeitet und was bereits abgeschlossen ist und was noch ausständig ist.