SCRUM-Dokument

rlm1-04: Interactive Room Map Display

Studiengang:	Informatik
Autor:	Marc Rudolf Fuhrer, Nicolas Anthony Waser, Nicola Sacha Eigel
Betreuer:	Prof. Dr. Michael Röthlin
Datum:	07.11.2022

Inhaltsverzeichnis

1 Versionshistorie

Version	Datum	Autor	Änderung
1.0	07.11.2022	Alle	1. Fassung «rlm1-04 - InteractiveRoomMapDisplay - SCRUM-Dokument - EigelFuhrerWaser.docx»
…			

2 Ausgangslage

2.1 Situationsbeschrieb

Begonnen bei den ursprünglich genannten Anforderungen, wurde das Team rlm1-04 dazu angehalten eine Web-Applikation mit definiertem Technologie-Stack zu entwerfen, deren Ziel es war Messdaten dreier anderer rlm1-Teams auf selbstentworfenen Gebäudekarten (mindestens SIPBB) und eigenem Koordinatensystem zusammenzutragen und einem BFH-Besucher nutzenbringend zur Verfügung zu stellen.

Nach einigen vertiefenden Absprachen mit dem Product Owner und sämtlicher beteiligter rlm1-Teams wurden diese Anforderungen sowohl konkretisiert, deren Fokus geshiftet, als auch teilweise ganz erneuert.

Stand der Ausgabe dieses Dokuments lautet die Ausgangslage des rlm1-04-Teams wiefolgt:
Zu entwerfen ist ein «Interactive Room Map Display» Service (genannt Service), welcher auf einem Low-Level-Interface fusst. Der Service ist wie angedacht webbasiert und bietet drei Schnittstellen:

1. Konsole
2. REST API
3. Browser-basierter Web-Zugriff

Sämtliche von uns abhängige rlm1-Teams gelten als «Publisher», da Sie Ihre Mess-Daten und die gesamte Darstellung als HTML über die REST-Schnittstelle publizieren.
Benutzern, genannt «Subscribers», soll es möglich sein, sich für Ihre gewünschten Gebäudeteile einzutragen und die damit verbundenen Informationen in folgenden Formaten zu konsumieren:

1. HTML (Darstellung in Browser)
2. JSON (Weiterverarbeitung)
3. PDF (Download)

Weiter ist der Zugriff über statische URIs zu gewährleisten, die Geschwindigkeit des Bildwechsels wählbar zu halten und Metadaten zu den Inhalten bereitzustellen.

Der Fokus des rlm1-04-Teams liegt auf der REST API, der Verifizierung der von Publishern bereitgestellten Daten und der ordentlichen Rechteverwaltung.

2.1.1 Problem

Zuständigkeiten innerhalb rlm1-Teams, Zeitdruck und begrenzte Verfügbarkeit dürften sämtliche Beteiligten vor eine Herausforderung stellen.

2.1.2 Chance

Eine Kompromissfindung zwischen PO, rlm1-Teams und rlm1-04-Team, welche den Erfolg einer jeden Partei, nicht zuletzt dem direkten und indirekten User des Services garantiert.

2.1.3 Zwang

Innert 5-wöchiger Frist war die Veröffentlichung einer versionierten API zu gewährleisten, welche unter Berücksichtigung der damaligen Anforderungen von Statten ging.

2.2 Stakeholder

2.2.1 PO (Product Owner)
Der PO, der sowohl als Repräsentant der projekt-bewertenden Instanz, als auch Vertreter der Interessen des effektiven Endbenutzers und Unterhalters des Endprodukts fungiert, ist daran interessiert eine ausbaufähige Lösung zur Live-Erfassung und Darstellung der Kombination von Gebäudeplänen und verschieden gearteter Sensor-Daten als Produkt dieser Arbeit zu sehen.
Er wünscht einen Service, der auf bewährten PHP-Komponenten basiert und einfach zu installieren, konfigurieren, bedienen und warten ist. Um eine gewisse User Experience zu gewährleisten soll sowohl eine vernünftige User- und Rechteverwaltung vorhanden sein, als auch eine vereinfachte Fehlerbehandlung durch freizügiges Logging umgesetzt sein.

2.2.2 rlm1-04-Entwicklerteam
Das Entwicklerteam möchte eine Software-Umsetzung, die möglichst sämtliche an Sie herangetragenen Anforderungen erfüllt, oder Sie bei allfälliger Weiterführung des Projekts über den Umfang des Moduls hinaus, durch Weiterverfolgung des gewählten Pfades, zu erfüllen vermag.

2.2.3 Service Administrator
Der Service Administrator wünscht sowohl umfassende User- und Publisher-Konfigurationsverwaltung, als auch die Verwaltung von Beziehungen dieser beiden Teilnehmer per Konsole. Weiter soll die Subscriber-Verwaltung restriktiv über IP-Addressbereiche (inkl. Wildcard) und Authentisierung umgesetzt sein. Auch alternative Zurverfügungstellung von Inhalten für unangemeldete User soll umgesetzt sein. Desweiteren sind Nutzungsdaten von Publishern und Subscribern zur Optimierung gefordert.

2.2.4 Publisher
Die Publisher sind darauf angewiesen, dass eine praktikable Schnittstelle zum Service vorzufinden ist, bei der Sie ausreichend Freiraum bzgl. vollständiger Übermittlung Ihrer Daten und deren Sicherheit haben. Realisiert wird dies über PUT-Requests via REST API per vom Service Administrator erstellte User(-Gruppen). Ordentliche http-Statusmeldungen sind unabdingbar. Sorgfältige Prüfung von Berechtigungen bzgl. Überschreiben von Daten anderer Publisher ist ebenfalls Voraussetzung. Ein Publisher möchte auch wissen, wie oft seine Daten abgerufen wurden, um seine Nutzung des Services sukzessive zu verbessern. Vorerst begnügt sich ein Publisher mit einem Low-Level-Interface, welches ihm erlaubt eine View in Form einer HTML-Seite zur Verfügung zu stellen. In einem weiteren Schritt ist ein High-Level-Interface angedacht, wie es zu Beginn des Moduls angedacht war.

2.2.5 Subscriber
Die Subscribers nutzen den Service, um sich die von den Publishern veröffentlichten Daten und die dazugehörigen Meta-Daten anzuzeigen. Subscribers sollen authentifiziert werden und möchten selbst bestimmen welche Informationen sie abonnieren möchten. Diese werden auf drei Arten abgerufen:

1. HTML, um Sie in einem Webbrowser mit selbstgewählter Frequenz als Slideshow anzuzeigen
2. JSON, um Sie in anderen Applikationen weiterzuverwerten
3. PDF als Download

2.3 Hypothesen

2.3.1 Hypothese 1
Eine der zentralen Erfolgsgaranten dürfte die Abwägung sein, ob die Zusammenarbeit der rlm1-Teams koordiniert (vgl. mit Release-Trains) erfolgen sollte, oder ob sich die funktionalen Anforderungen so weit herunterbrechen lassen, dass sowohl ein solides Fundament, als auch eine autonome Arbeitsweise der einzelnen rlm1-Teams zum Erfolg des Services führt.

2.3.2 Hypothese 2
Unterteilung in Low-Level und High-Level-Interface ist dem agilen Charakter im Sinne des agilen Manifests bzgl. «Stetige Zusammenarbeit mit Kunden über Vertragsvereinbarungen» zuträglich.

3 Produktziel

3.1 Produktziel Inkremente zu Sprintende

Diesbezüglich wurden dem rlm1-04-Team weitläufig Freiheiten eingeräumt. Es existieren lediglich zwei relevante Abgaben des Services (API-Spezifikation nach Woche 5, Schlussabgabe Service) und zwei Präsentationen im Plenum. Ein beständiger Austausch mit dem PO wird garantiert.

3.2 Produktziel 5 Wochen ab Projekt-Start

REST API Spezifikation (wurde erfüllt, jedoch durch neue Anforderungen abgelöst)

3.3 Produktziel Schlussabgabe

Der Service als Ganzes soll auf dem Webserver p1-irm.bfh-web-labs.ch erfolgen und ist, geschuldet dem weit verbreiteten Technologiestack (PHP/Laravel), 1-zu-1 übertragbar auf sämtliche gängige Umgebungen.

3.4 Produktziel Laufend

Sämtliche Prozess-Aktivitäten sind über Jira/Confluence nachvollziehbar (Sowohl Entwicklung, als auch Management). Sämtliche Besprechungen werden protokolliert und mit Confluence bereitgestellt.

- https://niceigel.atlassian.net/

Gewährleistung der Nachvollziehbarkeit des Quellcodes ist über das mit Jira verknüpfte Github-Repository garantiert. Bei jedem Push erfolgt die Auslieferung an den Kunden und automatisches Deployment. Es wurde von abhängigen Teilnehmern darauf verzichtet Stable-Releases parallel zu erhalten. Nach Anforderungs-Update «Low-Level-Interface» wurde Einigung erzielt über versionierte API auf produktive Umgebung Zugriff zu erhalten.

- https://github.com/marcfuhrer-com/rlm1-04

Desweiteren bietet das rlm1-04-Team eine umfassende Gesamtspezifikation mit Anforderungsanalyse, Modellierungen, etc. in Form eines OneDrive-Shares an.

- Project1 (rlm1-04) - Documentation and Models

4 Scrum-Rollen

4.1 PO Product Owner (=Service Owner)

Prof. Dr. Michael Röthlin

4.2 Scrum Master

Nicola Sacha Eigel, zertifiziert

4.3 Developer

Nicola Sacha Eigel, Marc Rudolf Fuhrer, Nicolas Anthony Waser

4.4 Service Administrator

Zur Zeit fiktiv. Verantwortlich für Administration der Systemkonfiguration

4.5 Publisher

Teilweise fiktiv. Messung & Aufbereitung bis Bereitstellung View für Subscriber. Zur Zeit Entwickler-Teams rlm1-01, rlm1-02 und rlm1-05.

4.6 Subscriber

Zur Zeit fiktiv. Zeigt ausgewählte Gebäudeteile auf Gerät mit Screen

4.7 User von Gebäuden und Räumen

Konsumenten von öffentlichen Bildschirmen, Computer oder mobilen Endgeräten

5 Sprintziel

5.1 P1 Sprint 1

- Datum: 07.10.2022 – 18.10.2022
- Sprint-Ziel: Alle Stories abgeschlossen, gereviewed und Grundstein für Entwicklung gelegt.
- Estimated Story Points: 34
- Achieved Story Points: 34

5.2 P1 Sprint 2

- Datum: 20.10.2022 – 01.11.2022
- Sprint-Ziel: Fixing vom Scope. Planen und Dokumentieren Features, Anforderungen usw. Konzepte für Software vorhanden damit man bald anfangen kann mit Umsetzung.
- Estimated Story Points: 34.5
- Achieved Story Points: 25.5

5.3 P1 Sprint 3

- Datum: 03.11.2022 – 14.11.2022
- Sprint-Ziel: - Alle nötigen Stories für die Zwischenpräsentation abgeschlossen - Zwischenpräsentation als Hauptziel - Einpflegen der neuen Anforderungen in die diversen Artefakte - Soweit wie möglich Abschliessen der Dokumentationen - ACL Konzept - Bereit für Umsetzungen nach diesem Sprint
- Estimated Story Points: (laufend)
- Achieved Story Points: (laufend)

6 Anforderungen

6.1 Epics

6.1.1 P1-18 REST-API
- Zeitraum: 01.10.2022 – 15.11.2022
- Status: In Arbeit

6.1.2 P1-19 Entwicklungsumgebung einrichten
- Zeitraum: 01.10.2022 – 18.10.2022
- Status: Fertig

6.1.3 P1-21 Frontend mit Map-Darstellung
- Zeitraum: 13.12.2022-10.01.2023
- Status: Zu erledigen

6.1.4 P1-22 Verwaltungs/Admin-Zugang (Anforderung abgelöst)
- Zeitraum: 30.10.2022 – 03.11.2022
- Status: Anforderung abgelöst

6.1.5 P1-62 Gesamtspezifikation
- Zeitraum: 01.10.2022 – 10.01.2022
- Status: In Arbeit

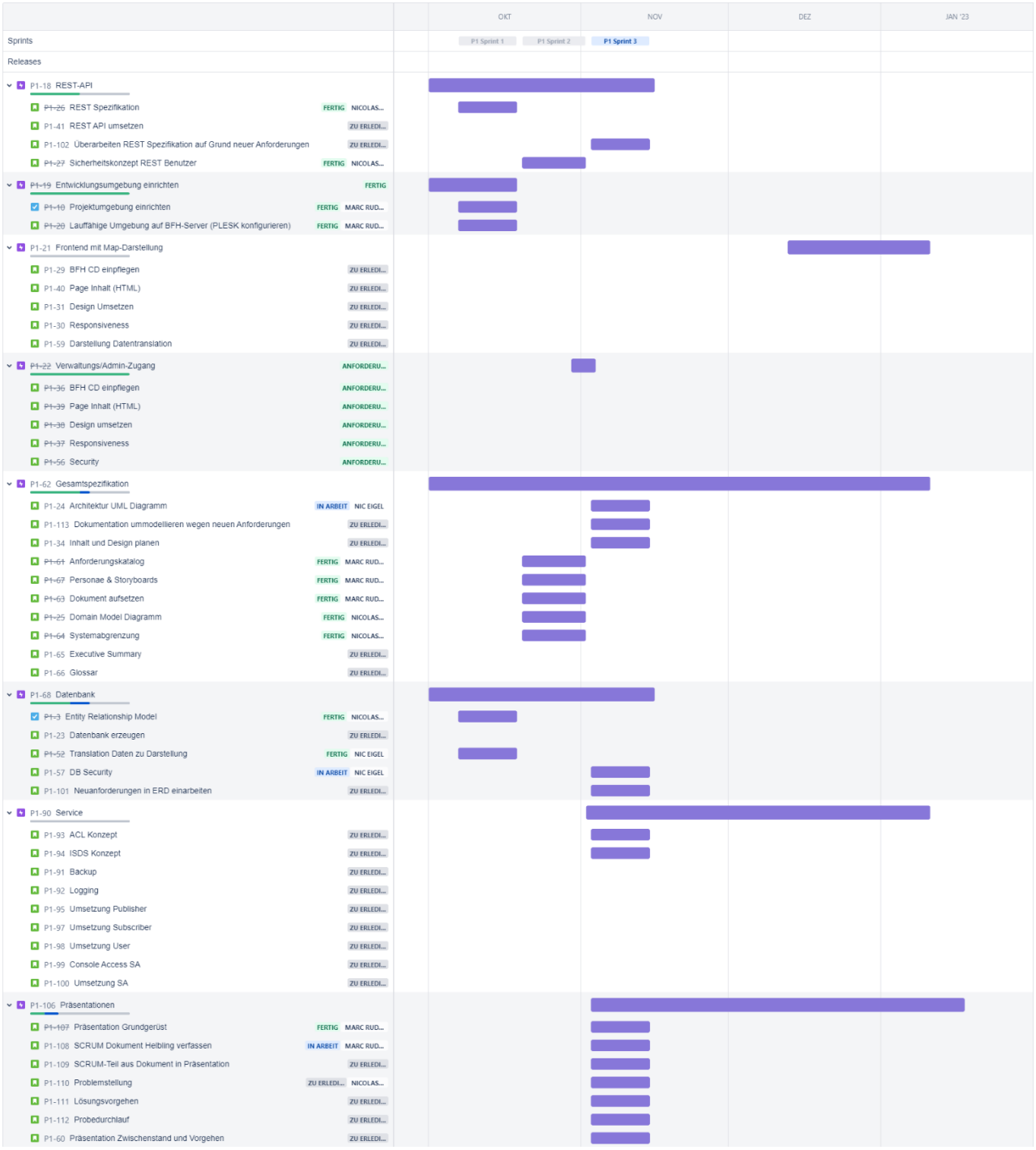
6.1.6 P1-68 Datenbank
- Zeitraum: 01.10.2022 – 15.11.2022
- Status: In Arbeit

6.1.7 P1-90 Service
- Zeitraum: 02.11.2022 – 10.01.2023
- Status: Zu erledigen

6.1.8 P1-106 Präsentationen
- Zeitraum: 03.11.2022 – 17.01.2023
- Status: In Arbeit

6.2 Anmerkung Backlog

Sämtliche Tasks werden den Epics zugeordnet. Beim jeweiligen Sprint-Planning werden Estimated Story Points vergeben und sichergestellt, dass jeder Task eine DoR/DoD aufweist. Jeder Entwickler wird dazu angehalten sinnvolle Sub-Tasks zu generieren, die der Nachvollziehbarkeit dienen. Mindestens ein Review wird zur Umsetzung des 4-Augen-Prinzips erwartet und als Sub-Task aufgenommen (Ergebnisse als Kommentar in Task).



Stand: 06.11.2022, 02:40 Uhr

7 Scrum Adaptionen

7.1 Daily SCRUM (StandUp)

Umgesetzt durch Confluence-Dokument. Zu Beginn wurden jeden Tag (auch bei Inaktivität) sämtliche Bestrebungen den Service voranzutreiben aufgeführt. Dies wurde durch die Verbesserungsvorschläge der Sprint-Retrospektiven auf effektive Arbeitsleistungen beschränkt. Ergänzend wurde der Austausch über einen WhatsApp-Chat als förderungswürdig befunden.

7.2 Weekly SCRUM

Seit Anbeginn des Projekts wurde der Dienstag ausgewählt (mindestens 15:00-16:00) – Kanal MS Teams. Dieser Termin erwies sich auch bezüglich wiederholt angenommener Einladungen an den PO Prof. Dr. Michael Röthlin als beständig. Sprint-Termine werden wann immer möglich auf Weekly gesetzt. Protokollierung selbstverständlich.

7.3 Spare Termin

Donnerstag abends (20:00 Uhr aufwärts) wurde jeweils als Reserve freigehalten, da sich dies als beste Möglichkeit neben beruflicher Tätigkeiten der Entwickler erwies. Protokollierung selbstverständlich.

7.4 Epics, Tasks, Sub-Tasks

- Rasch greifbare Planung über mehrere Ebenen
- Konsequente Anwendung von DoR/DoD auf Tasks und Sub-Tasks
- Story-Points spätestens bei Sprint-Planning erfasst
- Zustände: «Zu erledigen», «In Arbeit», «Fertig», «Anforderung abgelöst»
- Zuweisungen und Fälligkeiten
- Rege Nutzung der Kommentar-Sektion und Markierung mit Mail-/Push-Benachrichtigungen

7.5 Task-Reviews

- Konsequente Reviews eines jeden Tasks, um das 4-Augen-Prinzip zu garantieren

7.6 Sprints

- Gängige Sprint-Länge von 1.5 bis 2 Wochen
- Sprint-Reviews dokumentiert
- Retrospektive dokumentiert
- Umsetzung Retrospektive validiert
- Akkurates Sprint-Planning

7.7 Tools

Getreu State of the Art wurden folgende Tools erlernt und aufgesetzt:

7.7.1 Jira

Doppelte Buchführung durch Projekt-Management-Tool «Jira».

- Ein personalisiertes Board für die Projektarbeit inkl. Epics, Tasks, Subtasks, Berichte, Erwähnungen, Beobachtern, Kommentare, Story Points, Linksammlungen, Verknüpfungen sämtlicher Ressourcen ausser WhatsApp.
- Ein personalisiertes Board für Management / Administration
- Jederzeit Einsicht von PO möglich

7.7.2 Confluence

- Sämtliche Besprechungsprotokolle
- Zugangsdaten
- Daily SCRUM

7.7.3 Github

- Integration mit Jira (Feature-Branches automatisiert, Zuweisung zu Tasks durch Commit-Messages)
- Integration mit BFH-Server (automatisches Deployment bei Push auf Main-Branch)
- Abstimmung Scripts für Uniformität Handhabung Deployment auf lokalen Entwicklungsumgebungen

7.7.4 Draw.io (PlugIn for Jira)

- Vereinfachte Nachbearbeitung bei Anforderungsänderungen
- Direkte Verknüpfung von Tasks

7.7.5 OneDrive

- Gesamtspezifikation nach BFH CD (abgeleitet von Requirements Engineering) - versioniert
- Modell-Exporte – versioniert
- Präsentationen