18.02.2020 // Hackathon, 07.02.20

Strukturiert nach: https://www.mendix.com/de/blog/eine-praktische-anleitung-fur-den-product-canvas/

Product Vision Board

- Warum erstellen wir die Anwendung?
  - Wir wollen ein System haben, das auf die Bedürfnisse offener Werkstätten zugeschnitten ist.
    - Veränderbar
    - bezahlbar
  - Denn bestehende Systeme sind
    - … zu teuer
    - … passen nicht vollständig zu unseren Bedürfnissen.
    - … nicht problemlos in bestehende Strukturen integrierbar
- Wer ist unsere Zielgruppe?
  - Offene Räume
    - HackerSpaces
    - Offene Werkstätten
  - FabLabs in/an Hochschulen
- Was sind ihre Bedürfnisse?
  - Möglichst autonomes / autarkes Handeln ermöglichen
    - selbstbestimmtes Arbeiten und Lernen
    - Einfacher niederschwelliger Zugang zu…
      - Türen
      - Maschinen
      - Werkzeuge
  - Einfache Administrationsprozesse
  - Die Möglichkeit haben, das System an individuelle technische Gegebenheiten anzupassen.
  - Möglichkeit zur Abrechnung dieser Vorgänge
  - Zugriff auf Monitoring / Protokollierung / Nutzungs-Statistiken
  - Granulare Berechtigungen
    - Maschinen: Nutzungsberechtigungen
    - Türen: Zutrittsberechtigungen
  - (Anleitung und Nutzungshilfen)
  - **Lösen von folgenden Problemen…**

        - manuelle zugangskontrolle
        - manuelles Aushändigen von Werkzeugen
        - verschiedene Authentifizierung methoden
        - manuelles Abrechnen
        - manuelles Buchführen von Nutzung und Verbräuchen
        - funktionierende Software/Hardware zu teuer (Fabman)
        - Nutzung von selbstgebauter Hardware aus versicherungstechnischen Gründen nicht möglich
- Was sind dabei unsere Ziele?
    - Das System soll sich verbreiten, das bedeutet.
        - Das System soll auf einer möglichst breiten und agilen Basis entwickelt werden.
        - Wir wollen für die User entwickeln, nicht für uns.
        - Das System soll Menschen Arbeit abnehmen statt aufbürden.
    - Das System soll im Rahmen einer nachhaltigen Struktur entstehen - nicht an einer Person hängen, sondern jederzeit durch andere weiterentwickelt werden können.
    - –> strictly OpenSource.
    - In Zukunft optional als Föderation funktionieren

Abgrenzung

Zur Diskussion

- Spaces die sich neu einrichten (Hauptzielgruppe sind bestehende Labs)
- Spaces mit homogenen Geräte/Maschinenpark (Hauptzielgruppe hat eine organisch gewachsenen Gerätepark)
- Spaces ohne Geräte/Maschinen (Hauptzielgruppe haben diese)
- Spaces die Geräte und Werkzeuge verleihen (Hauptzielgruppe hat Werkzeuge immer am Ort)
- Spaces mit viel Geld (Hauptzielgruppe hat knappes Budget)
- Spaces die Service möchten (Hauptzielgruppe kann sich selbst helfen)
- Spaces die eine Einzellösung benötigen

Fragen zur Zielgruppe

- Darf unsere Zielgruppe Veränderungen an Maschinen vornehmen?
- Darf unsere Zielgruppe Veränderungen an Türen vornehmen?
- Darf unsere Zielgruppe virtuelle Guthaben verwalten?
- Hat die Zielgruppe feste ein Buchungsprozess (vorherige Vergabe fester Zeitslots) für Geräte

- Wie hoch ist die Fluktuation von Endnutzer unserer Zielgruppe?
- Wie hoch ist die Fluktuation von Technischen Administratoren unserer Zielgruppe?
- Wie hoch ist die Fluktuation von Supervisor unserer Zielgruppe?
- Kurz wie viel Einarbeitung von allen Personas ist noch zulässig?
- Welche Skills und technischen/nichttechnischen Möglichkeiten stehen der Zielgruppe in welchen Umfang zur Verfügung?
  - Techniker (Kabel ziehen, Geräte modifizieren)
  - Netzwerker (Netzwerke einrichten, Systeme warten, Backups durchführen und zurückspielen)
  - Einweiser für Endnutzer (Karten ausgeben, Ansprechpartner)
  - Einrichter (Einrichten des Systems, Probleme beheben)
  - Entwickler (Erweiterung und Anpassung des FabAccess) ? welche Sprachen sind zu erwarten?
- Welche Geräte sind zu erwarten?
  - 3D Drucker
  - CNC Fräsen
  - Plotter
  - Drucker
  - Nähmaschinen
  - Lötkolben
  - Werkzeugkoffer
- Wie viele und welche Besonderheiten haben die Räume unser Zielgruppe?
  - Anzahl von Räumen?
  - Sind Räumlichkeiten verteilt auf mehrere Standorte?
  - Haben die Türen unterschiedliche Öffnungsmechanismen?
  - Können Öffnungsmechanismen getauscht werden?
  - Gibt es Sicherheitstüren?
  - Gibt es kritische Bereiche?

Product Canvas

Personas

- **Wer sind unsere Nutzer?**
  - Akteure, die sich Zutritt zu Dingen verschaffen (Nutzer*, Mitglieder der Werkstatt / des Raumes)
  - Der technischen Administrator (installiert / aktualisiert das System Software/Aktoren/…)
  - Supervisor (communitynaher Administrator)

- Finanzadministrator
- Ressourcen-Administrator
- Technischer Maschinen-Betreuer
- Lab-Lead (Ansprechpartner in der Räumlichkeit)

User Journeys

- **Was sind Eigenschaften sowie Aufgaben der Nutzer und wie stellen wir uns vor, dass sie diese erfüllen?**
  - Nutzer / Akteure, die sich Zutritt zu Dingen verschaffen
    - Eigenschaften
      - sind tendentiell ungeduldig, möchten schnell zu Ergebnissen kommen
      - sie sind Laien oder technische Spezialisten - volle Bandbreite
      - sie möchten jederzeit in die Räume und darin arbeiten
    - Aufgaben
      - sollen den Arbeitsschutz beachten (sind daran aber nicht wirklich interessiert)
      - sollen auf die Maschinen achten (verschleißarme Anwendung)
      - sollten aufräumen (wollen es aber nicht)
      - Registrierung: gibt seine Daten selbständig ein und stimmt deren Verwendung zu
    - Bedürfnisse
      - wollen den Preis der aktuellen Verbrauchszeit wissen
      - wollen fehlgebuchte Buchungen stornieren
      - wollen eine Monatsübersicht
  - Technischer Administrator
    - Aufgaben
      - Will / soll das System (die Software/Aktoren/…) installieren
      - sowie aktiv halten (aktualisieren, Fehler beheben)
    - Eigenschaften
      - kennt das System, kann Software installieren
      - kann nicht zwangsläufig programmieren
      - “Semi-Nerd”
    - Infrastruktur
      - für die Software steht ihm entweder ein eigener Server zur Verfügung oder auch nicht
    - Bedürfnisse
      - will von bestehenden Geräte-Beständen Eigenschaften importieren / exportieren
      - hat ggf. bevorzugte Authentifizierungsmethoden

        - moechte moeglichst wenig Administrationsaufwand
        - möchte / kann nicht viel Zeit in das Aufsetzen investieren
        - Möchte nicht ständig Angriffsvektoren abschneiden oder schließen (müssen).
        - will nicht, dass Module bei einem Update kaputt gehen
- Supervisor (Nutzer*-Administration)
    - Aufgaben
        - Vergibt z.B. Berechtigungen nach Workshops oder Einweisungen
        - Empfängt neue Mitglieder, vergibt Berechtigungen
        - Kann Nutzerdaten eingeben und ändern
- Finanz-Administration
    - Aufgaben
        - rechnet gegenüber den Nutzern ab
    - Bedürnisse
        - er möchte individuelle Verbrauchswerte erfassen
        - es sollte unterschiedliche Typen von “Rechnungen” geben. Nicht jeder Nutzer braucht eine volle Rechnung mit MwSt.
        - möchte seine individuelle Abrechnungssoftware nutzen
- Ressourcen-Administration
    - Aufgaben
        - Kauft Material ein
        - Legt Material in Lager
        - gibt Geld aus
    - Bedürnisse
        - Brauch Verbrauchsdaten von Material konsumierenden Maschinen (z.B. 3D-Drucker)
- Technischer Maschinen-Betreuer
    - Aufgaben
        - kümmert sich um Maschinen
    - Bedürfnisse
        - Verschleißdaten / Nutzungsdaten von Maschinen sollten angezeigt werden können: Rohdaten!
- Lab-Lead (Ansprechpartner in der Räumlichkeit)
    - Aufgaben
        - Hat Überblick über den aktuellen Betrieb im Labor
        - Können einfache Fragen zur Nutzung beantworten.
    - Bedürnisse
        - will wissen wer gerade da ist
        - wissen welche Maschinen aktiv sind
        - will Nutzer* und ihre Eigenschaften recherchieren können

        - will einfach Stornierungen vornehmen, wenn Falschbuchungen vorliegen
    - Eigenschaften
        - sind *immer* vor Ort am Tresen
        - sind freundlich
        - haben z.T. geringes technisches Verständnis vom System
- Programmierer
    - Schreiben den Kern
    - Wollen realisierbare Aufgabenpakete lösen
    - Hören auf die (anderen!) Nutzer

Technische Lösungen

- Codearchitektur (abgeleitet aus den oben stehenden Bedürfnissen)
    - Es gibt einen Kern mit Funktionen / Methoden, die den kleinsten gemeinsamen Nenner bilden
        - Kontrolliert Zugangsberechtigung
            - Von außen kommt eine Anfrage, die wird validiert und die Entscheidung mitgeteilt
                - Authentifikation
            - Parameter, die die Entscheidung beeinflussen, können vom Administrator über ein externes Interface angepasst werden
        - Der Kern hat ein integriertes Protokollsystem
            - z.B. Zeitspempel der Authentifizierung eines Users / Aktivierung eines Aktors / Grund der Ablehnung von Aktion XY
            - gibt Rohdaten (z.B. auf Socket) aus.
        - Gibt generisches “an” oder “aus” heraus. Was der “Aktor” oder “Sensor” damit macht, ist ihm überlassen
    - Um den Kern gibt es optionale Frameworks / Plugins, die Zusatzfunktionen ermöglichen, wie:
        - konkrete Authentifizierungsmethoden
        - Aktoren-Definitionen
        - Schnittstellen zu ERP-Systemen
        - Rohdaten der Protokollierung werden verarbeitet und abgelegt / ausgewertet / whatever.
    - **Begründung:** Ein komplett offenes Framework bedeutet beim Einrichten sehr viel Arbeit (benötigt technisch versierten Administrator oder ein sehr aufwendiges Admininterface) und da das höchste Ziel die Benutzerfreundlichkeit und geringen Kosten (auch Zeit) sind, leitet sich die Entscheidung daraus ab.

To-Dos:

- Technische Anforderungen klarer fassen
- Eine Umfrage formulieren mit den folgenden Parametern:
  - Wer ist bereit, als Alpha-Tester zur Verfügung zu stehen?
  - Verbreitete Authentifizierungsmethoden (RFID/NFC/UserId+Pwd/...?)
  - Bereits existierende Nutzerdatenbank (LDAP, SQL, AD, ...?)
  - Bereits existierende Schliess-/Zugangssysteme
  - Verwendete ERP-Systeme
  - Verbeitete Aktoren
    - 3D-Drucker? Welcher?
  - Welche Server werden verwendet?

Zusammenfassung 08.02.

- Wir wollen agil arbeiten uns aber keine Rituale mit massivem Kommunikations-Overhead aufbürden
  - In Vollzeit-Teams ist der Kommunikations-Overhad vernachlässigbar. In unserer Konstellation (4 Stunden pro Woche) eher tödlich.
- Asynchrone Kommunikation ist wichtig und in unserem Fall hilfreich - das lässt manche Elemente der Agilen Arbeitweise nicht zu.
- Jira können wir testen und schauen ob es uns hilft oder Mailinglisten bzw. die in Gitlab vorhandenen Möglichkeiten sinnvoller sind.
- Grobe Vorstellung der Arbeitsweise:
  - Es gibt Issues/Tickets in JIRA die das Ziel haben eine konkrete Entscheidung (architekturell) zu treffen
  - Diskussionen finden asynchron innheralb dieser Issues/Tickets statt
  - Ziel der Diskussion sollte bei architekturellen Entscheidung ein ADR sein (Architectual Decision Record)
    - ADR-Template.
    - CLI-Tool zur Erstellung und Organisation.
    - Erklärender Blog-Beitrag
  - Auslieferbare Features definieren wir in Stories
- Tasso richtet eine Mailingliste ein
  - Bis JIRA läuft findet die asynchrone Kommunikation über die Mailingliste statt
- Architekturskizze muss nochmal diskutiert und als Bild entwickelt werden.
  - Hierbei wollen wir auf die Architekturen der bereits im Team vorhandenen Backends eingehen, um zu schauen wo Synthese Sinn macht

---

Version #2
Erstellt: 13 Oktober 2024 00:47:26 von Mario Voigt (Stadtfabrikanten e.V.)

Zuletzt aktualisiert: 14 Dezember 2024 18:23:07 von Mario Voigt (Stadtfabrikanten e.V.)