Mit Lean UX zum nutzerorientierten Entwurf von Legal Tech Anwendungen

Beitragsreihe: LEGAL LIVE 2021 – Inhalte, Erfahrung, Feedback

Dr. Carsten Günther | Geschäftsführer Codefy GmbH

02. September 2021
LRZ 2021, Seiten 179 bis 186 (insgesamt 8 Seiten)

Dieser Beitrag ist eine Zusammenfassung des Vortrags von Dr. Carsten Günther am 23.03.2021 auf der LEGAL LIVE 2021 mit demselben Titel.

I. Nutzerzentrierte Automatisierung von Arbeitsprozessen

Mit Hilfe von Softwarelösungen werden Tätigkeiten und ganze Arbeitsprozesse vereinfacht und zum Teil sogar automatisiert. Dies gilt auch für textbasierte Arbeitsabläufe in Kanzleien und Rechtsabteilungen von Unternehmen. Gerade die Fortschritte im maschinellen Lernen und der Anwendung neuronaler Netze in den letzten fünf Jahren lassen nun die Entwicklung von Softwareprodukten zu, die eine Verarbeitung von gesprochener und geschriebener Sprache in großem Umfang und hoher Zuverlässigkeit gestatten. Komplexe Algorithmen, umfangreiche mathematische Modellierungen von Textarten – beruhend auf mit unvorstellbar großen Datenmengen vortrainierten Textmodellen – und eine flexibel erweiterbare Rechenleistung aus der Cloud erlauben eine große Funktionsvielfalt von Softwarelösungen. Heute könnnen gänzlich neue oder umfangreich erweiterte Anwendungen zur Suche im Dokumentenbestand einer Kanzlei, zur Klassifikation und Zusammenfassung von Texten, zum Knowledge Management in einer Organisation, zur maschinellen Übersetzung oder zur Textgenerierung umgesetzt werden.

Doch wir alle werden bereits die Erfahrung gemacht haben, dass uns die Funktionsvielfalt und Leistungsfähigkeit beim ersten Kontakt mit neuen Softwarelösungen sehr schnell überfordert. Das kann dazu führen, dass wir gar nicht die ganze Leistungsvielfalt eines Softwareproduktes ausschöpfen und uns stattdessen auf die Nutzung einiger weniger Funktionen beschränken oder das Produkt gänzlich ablehnen. Durch einen

nutzerorientierten Entwicklungsprozess kann man möglichen Problemen bei der Produkteinführung entgegenwirken.

Entscheidend ist, dass ausgehend vom ermittelten Bedarf einer Nutzergruppe Softwarelösungen konzipiert werden, die zielführend zu bedienen sind, den dringendsten Bedarf der Nutzer decken und ressourcenschonend die Algorithmen auswählen, die das jeweilige Problem lösen. Der letztgenannte Punkt ist zu beachten, da natürlich komplexe neuronale Netzwerkstrukturen, trainiert mit Millionen an Dokumenten, viele NLP-Probleme mit einer hohen Genauigkeit lösen – jedoch sowohl in der Trainingsphase als auch in der Anwendungsphase große Rechenleistung und hohen Speicherbedarf erfordern. Alternativ können Verfahren aus dem maschinellen Lernen zur Anwendung kommen, die für bestimmte Anwendungsfälle ähnliche, bisweilen schneller bessere Ergebnisse liefern und mit weniger Aufwand (Prozessor- und Speicherbedarf) zu trainieren sind.

II. Der Ablauf eines nutzerzentrierten Software Design- und Realisierungsprozesses

Selbst wenn man sich das Ziel setzt, eine nutzerfreundliche Software zu erstellen, ist man nicht davor geschützt, seine Lösung doch wieder durch eine reine Technikorientierung des Developmentteams mit Funktionen und Auswahlmöglichkeiten zu überfrachten und schlussendlich den zukünftigen Nutzer[1] zu überfordern.

Um das zu vermeiden, sind in den letzten Jahren eine ganze Reihe von formalisierten Methoden und Werkzeugen entwickelt worden, die den Entwurf, die Umsetzung und die Einführung von softwaregestützten Innovationen steuern. Es geht im Kern immer wieder darum, aus den oft unscharf formulierten Bedürfnissen der zukünftigen Softwarenutzer formalisiert Anforderungen an ein Softwareprodukt abzuleiten und diese dann strukturiert umzusetzen und einen messbaren Kundennutzen zu erzielen.

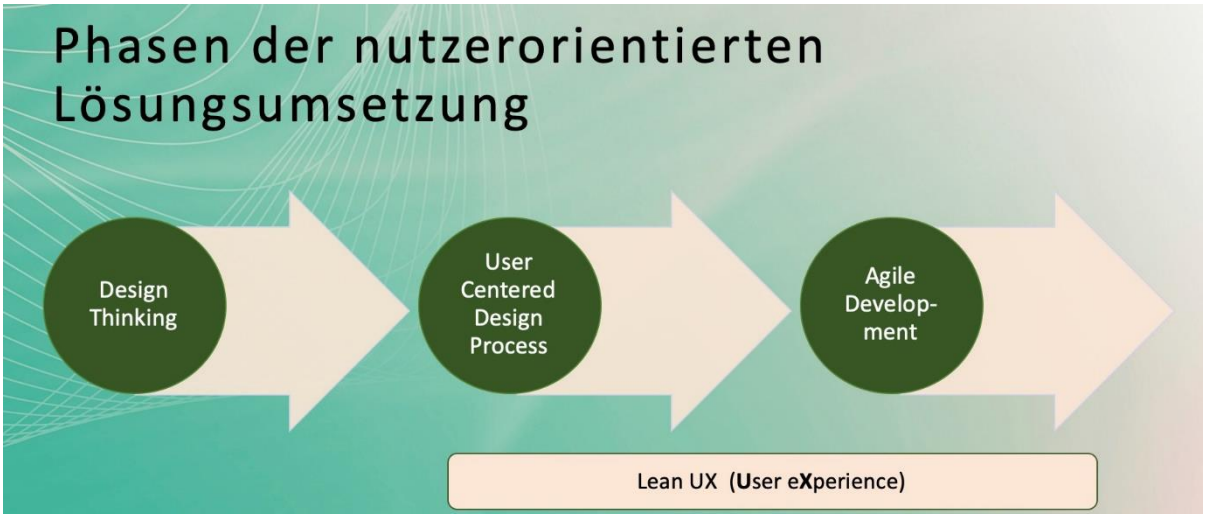
III. Drei Phasen – ein Ziel: Design Thinking + UX + Agile

Die nutzerorientierte Umsetzung einer Innovation – sei es ein neues Softwareprodukt oder ein neuer Software-basierter Service – kann man grob in drei Phasen unterteilen:

1. Strategische Orientierung über Design Thinking Workshops,
2. Nutzerorientiertes Design der Softwarelösung und einhergehende

[1] Aus Gründen der besseren Lesbarkeit wird auf die gleichzeitige Verwendung weiblicher und männlicher Sprachformen verzichtet und das generische Maskulinum verwendet. Sämtliche Personen- und Rollenbezeichnungen gelten gleichermaßen für alle Geschlechter.

3. Agile Entwicklung der Software.



Im ersten Schritt, der strategischen Ausrichtung eines Innovationsvorhabens, haben sich Design Thinking Workshops als ein probates Mittel etabliert. Diese Workshops folgen einem wohldefinierten Muster und gliedern sich in zwei Phasen:

1. der Analyse des **Problemraums** – d.h. des aktuellen Arbeitsumfelds und der Herausforderungen der Zielgruppe, aber konkrete Lösungsmöglichkeiten werden hier nicht diskutiert;
2. der Skizzierung des **Lösungsraums** – d.h. Entwicklung verschiedener Lösungsmöglichkeiten und Bewertung anhand der Kriterien aus der ersten Phase.

Gewünschtes Ergebnis eines Design Thinking Prozesses ist ein von potentiellen Nutzern bewerteter Prototyp z.B. in Form eines Interaktionskonzeptes – ohne dass eine tatsächliche technische Realisierung dieses Prototyps erfolgt. Die Design Thinking Methode wird für die ganze Bandbreite möglicher Innovationsvorhaben eingesetzt, natürlich auch für die Rechtsberatungsbranche. Hier sind diese Methoden unter dem Begriff des **Legal Design Thinking** auf die spezifischen juristischen Prozesse adaptiert worden, um insbesondere die Belange der Mandantschaft in den Fokus zu rücken.

IV. Mit Lean UX zum Erfolg

Auf der Grundlage dieses sehr abstrakten Prototypen muss anschließend die konkrete Realisierung einer Softwarelösung konzipiert und umgesetzt werden. Auch hier haben sich in den letzten Jahren formalisierte nutzerorientierte Methoden der Konzeption und der Umsetzung etabliert, die iterativ in mehreren aufeinander abfolgenden Zyklen ausgeführt werden. Kernpunkt ist auch hier wieder, dass sich die Arbeiten an den Bedürfnissen der zukünftigen Nutzer leiten lassen. Nutzerorientierte Konzeptionsarbeiten zum Design der Zielanwendung und in der Regel agil gesteuerte Softwareentwicklungsarbeiten wechseln einander ab. Inkrementell und iterativ werden

auf diese Weise die innovativen Ideen umgesetzt und der Nutzergruppe zur Erprobung übergeben.

Es werden aber nicht einfach nach Belieben Produktinkremente der Zielgruppe übergeben, sondern auch diese iterativen Schritte müssen einer abgestimmten Planung unterliegen, um zum Ziel zu führen. Ein zu planendes Zwischenergebnis einer Iteration auf dem Weg zur Softwarelösung bzw. zum Softwareprodukt ist ein MVP – „Minimum Viable Product". Dabei handelt es sich um eine minimale, aber schon brauchbare Produktversion, die dazu dient, schnellstens Nutzerfeedback zur bisherigen Umsetzung zu bekommen. Dadurch sollen die Ergebnisse der bisherigen Konzeptions- und Umsetzungsschritte verifiziert und die nächsten Schritte geplant werden. Dabei ist abzuwägen, dass einerseits das jeweilige Produktinkrement noch „minimal" ist, also nicht zu viel Entwicklungszeit bereits hineininvestiert worden ist. Aber andererseits muss die Umsetzung eines Produktmerkmals schon so funktionieren, dass man basierend auf der tatsächlichen Nutzung und nicht aus einer theoretischen Betrachtung heraus Feedback von Kunden bzw. Nutzern erhält.

Dieser iterative, nutzerorientierte und agile Konzeptions- und Umsetzungsprozess wird unter dem Begriff **Lean UX** zusammengefasst und gliedert sich in die beiden Teilprozesse:

1. Lösungs- bzw. Produktkonzeption unter Nutzung der **UCDP-Methode** (UCDP – „User Centered Design Process"), d.h. eines **nutzerzentrierten Designprozesses**;
2. Softwarerealisierung mittels **agiler** Projektsteuerungsmethoden. Ein bewährtes Vorgehensmodell ist hier z.B. der **SCRUM-Prozess** mit definierten Teamstrukturen, Entwicklungsmethoden, Meeting-Formaten und Entwicklungs-zyklen, die Sprints genannt werden.

V. Nutzerorientiertes Vorgehen beim Produktdesign mittels UCDP

Der nutzerorientierte Designprozess einer Softwarelösung wird mittels Methoden und Werkzeugen strukturiert, die teilweise schon in den oben beschriebenen vorgelagerten Design Thinking Workshops zur Anwendung kommen, nun aber viel feingranularer Nutzerverhalten und -bedürfnis und die technischen Realisierungsmöglichkeiten spezifizieren.

Der nutzerorientierte Designprozess durchläuft iterativ wiederholt diese Phasen:

- Verstehen,
- Explorieren,
- Entwerfen und
- Testen.

Diese Phasen sind mit konkreten Arbeitsschritten unterlegt und als Arbeitsergebnisse liegen jeweils methodisch klar definierte Dokumentationen vor:

- die Definition von **Personas**,
- die Ausformulierung von **User Stories**,
- deren eventuelle Zusammenfassung zu **Epics**,
- die Durchführung des **Customer/User Journey Mapping** und
- die Ableitung von **Use Cases**.

Im Folgenden sollen nun diese Arbeitsschritte genauer beschrieben werden:

1. Der erste Schritt des Desigprozesses für eine neue Softwarelösung besteht in der Definition potentieller, aber fiktiver Nutzer – **Personas** genannt. Dabei wird wie folgt verfahren: Mittels Umfragen oder kontextbezogener Interviews werden Bedürfnisse, Erfahrungen und Ziele von prototypischen Vertretern der Zielgruppe systematisch erhoben und als quantitative und qualitative Daten entweder in Fließtext oder auf Formularkarten dokumentiert. Die abstrahierten Informationen beschreiben Alter, Familienstand, Lebensumstände, digitalen Erfahrungsstand, Ziele und Wertvorstellungen der potentiellen Nutzer. Diese Daten sollen dabei helfen, durch bisherige Softwarelösungen unbefriedigte Nutzerbedürfnisse aufzudecken. Die Auswahl der Personas ist sehr sorgfältig zu treffen und es sollten Vertreter **aller** Rollen berücksichtigt werden, die zukünftig mit dem Produkt in Kontakt kommen werden. Dadurch wird man für alle Nutzer eine positive Nutzererfahrung mit der Software sicherstellen können und einer durchgängigen Nutzung in der gesamten Organisation den Weg bereiten.
2. Anschließend wird vor dem Hintergrund der definierten Personas und der Anwendungsdomäne eine **Customer** bzw. **User Journey** entworfen. Über einen Zeitraum (z.B. Arbeitstag), der sich aus dem Anwendungsszenario ergibt, modelliert man die vielfältigen Kontaktpunkte der Nutzer mit dem zu erstellenden System. Indem man den Nutzungszeitraum breit abdeckt, werden Kontaktpunkte mit dem zu erstellenden Produkt erfasst, die nicht im Hauptnutzungszeitraum liegen wie z.B. statt am Schreibtisch in der Kanzlei im Home Office oder während

der Anfahrt zum Mandanten. Sofort wird ersichtlich, dass diese Nebennutzungsszenarien zusätzliche Anforderungen an die Software wie an die Darstellungsform (Bildschirmgröße), die Bedienbarkeit (Smartphone-Touchscreen) oder die zu übertragende Datenmenge (Mobilfunkverbindung) stellen werden. Indem man bei der Lösungskonzeption auch diese Situationen berücksichtigt, wird sich ebenso in ungewöhnlichen Kontexten eine positive Nutzererfahrung einstellen. Weitere Anwendungsfälle können aber andere Betrachtungszeiträume bedingen, wie z.B. den Lebenszyklus eines Projekts. Das wäre bei einem Legal Case Management System oder einem Due Diligence Tool zutreffender, bei denen man dann speziell die Phasen der Fall-Anlage, der Betreuung und des Abschlusses betrachten wird.

3. Danach werden für jeden Kontaktpunkt mit dem Zielsystem nutzerrollenspezifisch **User Stories** beschrieben und den Phasen in der oben aufgestellten User Journey zugeordnet (**User Story Mapping**). Diese User Stories beschreiben einzelne Anforderungen an die Softwarelösung in einem **formal strukturierten Satz** aus Sicht genau einer zukünftigen Nutzerrolle. Als Struktur einer User Story hat sich die folgende Form etabliert:
   „Als [**User**] möchte ich [**etwas können**], damit ich [**etwas erreichen kann**], um nicht [**etwas anderes tun**] zu müssen."
   Wichtig ist, dass für alle zuvor definierten Personas rollenspezifisch User Stories herausgearbeitet werden. Ein konkretes Beispiel für eine User Story ist: *„Als Associate möchte ich mit einer vorgegebenen Formulierung nach ähnlichen Absätzen in Kanzleischriftsätzen suchen können, damit ich aus Altfällen die Argumentationslinie eventuell wiederbenutzen kann, um diese nicht selber neu aufbauen zu müssen."*
   Nicht vergessen werden darf der dritte Bestandteil einer vollständig beschriebenen User Story: die Festlegung der Akzeptanzkriterien, wann eine User Story als realisiert gilt. Damit wird vor Beginn der Umsetzung definiert, wann in dem hier vorgestellten nutzerorientierten Paradigma diese User Story einen Wert für den Kunden, den Nutzer darstellt.
4. Soll mit der Lösung z.B. ein komplexer Prozess softwareseitig unterstützt werden, so kann die Anzahl der Rollen und damit dann auch die Anzahl der User Stories sehr groß und unübersichtlich werden. Auch ein einzelner Arbeitsschritt kann eine komplexe Struktur aufweisen, die auf mehrere User Stories abgebildet werden muss. Zur besseren Planung und Strukturierung der Konzeption werden dann mehrere User Stories zu einer großen User Story, **Epic** genannt, zusammengefasst. Im oben erwähnten juristischen Arbeitsumfeld können z.B. verschiedene, in Verbindung stehende Teilarbeiten wie Zugriff auf das Kanzleisystem, das Vorschlagen von ähnlichen Formulierungen oder Ablegen von Zwischenergebnissen bei der Modellierung zu dem Epic „Abgleich mit Altfällen" zusammengefasst werden.
5. Nachdem nun die Softwarelösung grob konzipiert ist, können die ersten Umsetzungsschritte vom Softwareentwicklungsteam geplant werden. Dazu sind die User Stories aber noch zu allgemein formuliert. Daher werden die User Stories

für die softwaretechnische Realisierung in einzelne **Use Cases** zerlegt. Die Use Cases unterliegen auch wieder strikten formalen Vorgaben, denn aus diesen werden die konkreten Entwicklungsaufgaben abgeleitet. Auf diese Use Case Beschreibungen kann dann systematisch aus den Entwicklertickets heraus referenziert werden. Die Entwicklertickets (Backlog Tickets) werden in dem agilen Softwareprojektsteuerungssystem abgelegt und dann sukzessive abgearbeitet.
Zum Verhältnis von Use Cases zu User Stories bleibt zu erwähnen, dass einzelne Use Cases in verschiedenen User Stories zur Anwendung kommen können, ohne dass dafür immer wieder neue Use Cases definiert werden müssen. Beispiele wären die Passworteingabe oder die Ansicht einer Ordnerstruktur. Auch hat sich bewährt, eine Nomenklatur zur systematischen Bezeichnung von Use Cases (z.B. SW-Komponenten-abhängig) zu etablieren, die dann auch im Softwareticketsystem verwendet wird, so dass sich Product Owner, Scrum Master, UI-Designer und SW-Entwickler (Rollen aus dem agilen SCRUM-Prozess) schnell abstimmen können. Auch ist darüber sehr einfach ein komponentenabhängiges Projekt-Controlling einrichtbar.

Diese Phasen und Arbeitsschritte werden im Projektverlauf wiederholt durchlaufen, so dass die Softwarelösung zunehmend immer feingranularer spezifiziert und anschließend umgesetzt wird und so immer besser die Nutzererwartungen erfüllen wird.

Betrachtet man noch einmal den Schritt von User Story über Use Case hin zum konkreten Softwareentwicklungsticket, ist klar, dass die Konzeption der Softwarefunktion als User Story noch sehr unscharf und unvollständig beschrieben ist, als dass damit ein Softwareentwickler einen zielführenden Code schreiben könnte. Doch ebendiese Diskrepanz ist nach dem Lean UX-Ansatz beabsichtigt, denn damit wird der Startpunkt einer gewollten intensiven **Kommunikation** zwischen Product Owner, Designer, Entwickler und Anwender gelegt. Genau diese Diskussion ist der wesentliche Punkt der neuen Art der Produktplanung, wodurch die nutzerorientierte Ausprägung des zur Umsetzung anstehenden Produktmerkmals erfolgt. Die konkreten technischen Anforderungen, die Gestaltung und Einbettung werden erst in den nun erforderlichen Kommunikationsphasen mit normierten Meeting-Strukturen (Refinement Meetings, Sprint Planning u. Review Meetings, Stakeholder Interviews usw.) näher definiert. Auf diese Weise bildet sich sukzessive ein sehr gutes gemeinsames Verständnis über den Nutzen und die Realisierung einer Nutzeranforderung heraus. Mit der anfänglich groben, unvollständigen Formulierung von Nutzeranforderungen wird in der modernen, agilen Softwareentwicklung genau dem Umstand Rechnung getragen, dass Nutzer innovativer technischer Lösungen erst beim konkreten Umgang mit ersten Versionen des Produkts (MVP) genau beschreiben können, wie die technische Lösung ihr Nutzerbedürfnis tatsächlich befriedigen kann.

Fertiggestellte Softwarelösungsinkremente werden in definierten Zeitabständen dem Kunden bzw. Nutzer zur Erprobung vorgelegt. Der Produkt Owner sammelt das neue

Nutzerfeedback ein und steuert darüber die nächsten Konzeptions- und Entwicklungsschritte. Auf diese Weise schließt sich der Kreis der nach Lean UX-Prinzipien gesteuerten Lösungsentwicklung.