Der Nutzer als Mittelpunkt: Design Thinking in agilen Softwareentwicklungsprojekten

Design Thinking wurde schon oft entdeckt – von Architektur über Produktdesign bis hin zum Management kennt man es. In der Softwareentwicklung sucht man diesen innovativen Ansatz bisher vergeblich. Dabei lässt er sich ideal mit agilen Vorgehensmodellen wie Scrum verbinden, um nicht nur effizient, sondern innovativ zu entwickeln. Doch wie sieht eine solche Integration aus? Kann sie Erfolg haben?

Softwareentwicklungsprojekte werden immer agiler. Gewünschte Inhalte entstehen deutlich schneller, kürzere Rückmeldungsschleifen ermöglichen Flexibilität und Raum, um die Entwicklung dynamisch zu priorisieren. Der Product Owner sieht frühzeitig Ergebnisse und kann immer wieder neue Wünsche einbringen. Betrachten wir industrielle Softwareprodukte jedoch genauer, so sehen wir viel zu selten Lösungen mit einfacher, intuitiver Benutzerführung, ansprechendem Oberflächendesign und wirklich guten, innovativen Lösungen für die Probleme der Benutzer. Genau hier kann Design Thinking weiterhelfen.

Design Thinking liefert einen Ansatz, um durch den Nutzer inspirierte und evaluierte Lösungsideen kontrolliert zu entwickeln. Durch eine Integration in agile Softwareprojekte entstehen so vom Nutzer akzeptierte, ansprechende, zeitnah bereitgestellte und evaluierte Lösungen.

Zeitnah und akzeptiert – das klingt nach agiler Methodik. Tatsächlich ist Design Thinking nicht weit entfernt und doch sehr verschieden. Während etwa Scrum darauf abzielt, die vorgegebenen Ziele im Backlog effizient und zeitnah umzusetzen, hilft Design Thinking dabei, Probleme neu zu denken und bisher ungeahnte Lösungswege aufzuzeigen und so echte Innovationen zu ermöglichen.

Was ist Design Thinking?

Design Thinking ist ein Rahmenwerk mit drei Phasen (**siehe Abbildung 1**). In der ersten Phase – *Definieren und Verstehen* – lernt das Team seine Nutzer kennen, entwickelt ein tiefgreifendes Verständnis für sie, ihre Probleme und ihre Umgebung. Dabei ist es essenziell, mit den Nutzern direkt zu interagieren, ihre Bedürfnisse und Ziele zu erkennen, sowie Problemstellungen zu definieren.

Mit dieser Fragestellung können in der nächsten Phase *Lösungsideen* frei entworfen und in formlosen, einfachen Prototypen umgesetzt werden. Ein Prototyp kann aus simplen Zutaten wie zwei skizzierten Kreisen auf dem Whiteboard und Post-it-Zetteln bestehen – und so das Drag&Drop-Prinzip darstellen. Ob die zuvor identifizierten Probleme zu den erarbeiteten Lösungsideen passen, untersucht das Team gemeinsam mit den Anwendern in einer *Testphase*. Verwenden sie die Lösungen? Wenn ja, wie? Welcher Prototyp löst welches Problem?

Sind die Ergebnisse einer Phase nicht zufriedenstellend, kann das Team diese wiederholen, in andere Phasen wechseln oder sie überspringen. Diese dynamischen Iterationszyklen werden solange durchlebt, bis ein Prototyp soweit ist, dass auf dieser Basis ein Produkt oder eine Dienstleistung entwickelt werden kann. Das in den durchlaufenen Phasen neu gewonnene Wissen und die Erfahrung über die Beziehung zwischen den Problemen und möglichen Lösungen wachsen damit während des Projektes kontinuierlich.

Vier Paradigmen im Design Thinking

Vier Paradigmen von Design Thinking bieten Ansatzpunkte für eine Integration in die agile Softwareentwicklung. Dabei handelt es sich erstens um freies, divergierendes und konvergierendes Denken; zweitens um iterative, interdisziplinäre Zusammenarbeit; drittens um den Nutzer als Mittelpunkt und viertens um greifbare und evaluierte Ideen.

1. Divergierendes und konvergierendes Denken

„To have a good idea, you must first have lots of ideas." (Linus Pauling, zweifacher Nobelpreisträger)

Eine besondere Bedeutung kommt dem ersten Paradigma zu: Design Thinking unterscheidet sehr explizit zwischen Phasen (**siehe Abbildung 2**) des

- divergierenden Denkens, freies, offenes und kreatives Denken ohne Einschränkungen und Sachzwänge, sowie

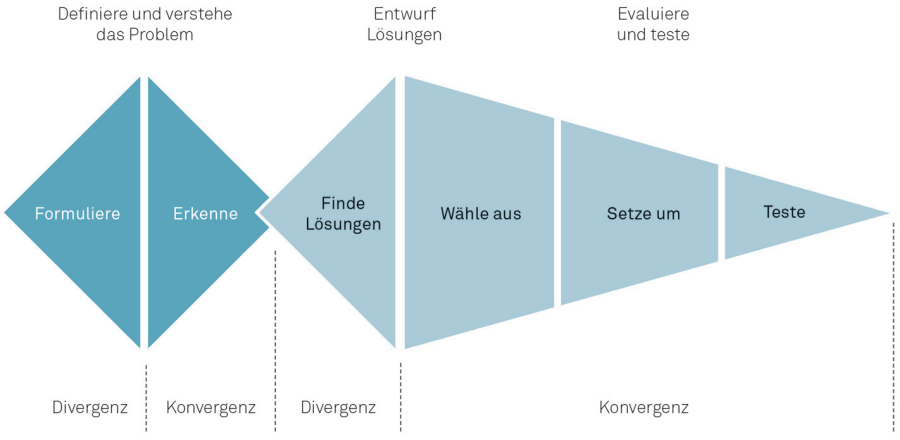

Abb. 1: Mögliche Ausgestaltung eines Design-Thinking-Prozesses.

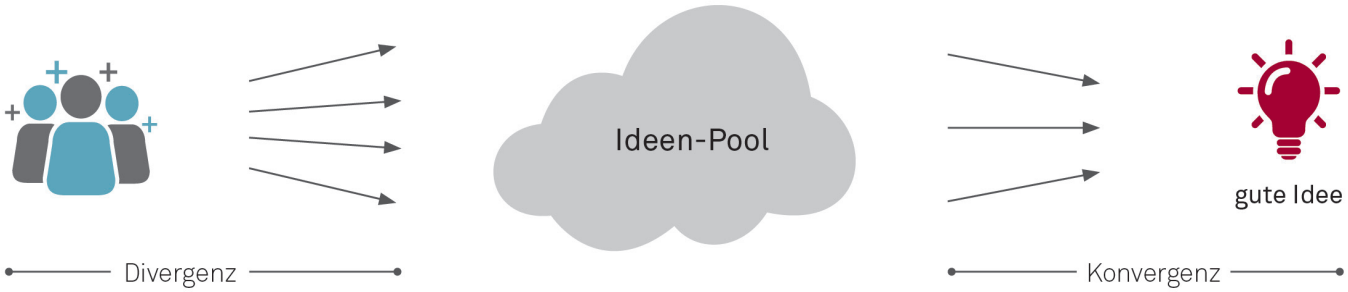

Abb. 2: Divergierendes und konvergierendes Denken.

- konvergierenden Denkens, also dem Treffen einer sinnvollen Auswahl.

Beide Phasen können und sollten wiederholt werden, denn die meisten Gedanken konvergieren irgendwann.
Während es in der divergierenden Phase also darum geht, möglichst viele Ideen zu generieren, geht es im konvergierenden Denken genau um das Gegenteil. Hier werden Optionen eliminiert, Probleme oder Ziele definiert und Entscheidungen getroffen. Tim Brown bezeichnet das kontinuierliche gegensätzliche Zusammenspiel dieser beiden gedanklichen Phasen als das „Saatgut" von Design Thinking [Bro09].

2. Interdisziplinäre Zusammenarbeit
Um neue, innovative Lösungsideen zu formen, ist es notwendig, eine Problemstellung aus sehr unterschiedlichen Perspektiven zu betrachten. Idealerweise besteht ein Design-Thinking-Team daher aus Charakteren unterschiedlichster Ausbildung, kultureller Hintergründe und mit verschiedenen Denk- sowie Herangehensweisen. Die Herausforderungen für ein Unternehmen im 21. Jahrhundert sind zu komplex, um sie einzelnen Abteilungen zu überlassen und Insellösungen zu schaffen.
Globale Kollaborationswerkzeuge wie das Web 2.0 oder Open-Source-Projekte sind perfekte Beispiele für die Bedeutung von interdisziplinären Teams [Gro09]. Die Zusammenstellung des Design-Thinking-Teams nimmt somit eine bedeutende Rolle im Design-Thinking-Ansatz ein.
Ein agiles Entwicklungsteam weist genau diese Eigenschaften auf – der Scrum Master kann Linguist sein, der erfahrene Mediendesigner sorgt für das Masken-Design, der promovierte Informatiker für die Systemarchitektur. Lediglich die Expertise der Anwender fehlt. Mit ihrer Unterstützung wären verständliche Dialoge, intuitive Workflows und innovative Lösungen, die Anwender effizient unterstützen und sogar begeistern, deutlich häufiger umsetzbar.

3. Der Nutzer als Mittelpunkt
„We watch what people do (and do not do) and listen to what they say (and do not say)." [Bro09]

Das Paradigma „Der Nutzer als Mittelpunkt" ist prägend für Design Thinking. Software wird von Menschen für Menschen erstellt – ob Benutzerschnittstelle oder nicht. Die Anwender entscheiden letztendlich, ob eine Erfindung tatsächlich zu einer Innovation wird. Deshalb werden sie nicht nur für Validierungselemente am Ende der Implementierung benötigt, sondern aktiv in die Entwicklung integriert. Der Benutzer steht zu jeder Zeit im Mittelpunkt allen Denkens innerhalb des Softwaredesigns und der Entwicklungsphase. Dies betrifft nicht nur, aber in besonderem Maße, die Oberflächenentwicklung, sondern auch das fachliche und technische Design.
Was bedeutet Benutzerzentrierung? Der Nutzer soll in den Lösungsfindungsprozess mit eingebunden werden und als eine Inspirationsquelle für neue Ideen dienen, jedoch diese nicht selbst erarbeiten. Ein negatives Beispiel ist eine einfache Befragung der Anwender: „Wie kann unser zukünftiges Supply Chain Management Sie am besten in Ihrer uns unbekannten täglichen Arbeit unterstützen? Kommen Sie mit dem System zurecht?" Dies generiert bestenfalls kleine und inkrementelle Ideen, kann aber definitiv keine Innovation hervorrufen, da der Nutzer meist gar nicht weiß, was er braucht.

4. Greifbare und evaluierte Ideen
Die entworfenen Lösungen sollen für die Nutzer jederzeit verständlich und konkret sein. Sie müssen die Lösungen begutachten und beurteilen können. Ein essentielles Paradigma von Design Thinking ist es daher, Ideen wirklich auszuleben. Eine Idee, die beispielsweise in Form eines Prototyps greifbar gemacht wurde, kann deutlich schneller evaluiert werden als eine nur verbal kommunizierte Idee. Nach der Beurteilung kann die Idee wieder angepasst oder bei Nichtgefallen sogar verworfen werden. Lösungen sollen schließlich immer durch den Nutzer evaluiert werden.
Beim Entwickeln von Prototypen geht zunächst Quantität vor Qualität. Je mehr Ideen ein Team während einer Entwicklung verwirft, desto mehr Wissen akkumuliert es zur Beziehung und den Einflussfaktoren zwischen den Problemen und ihren Lösungen. In späteren Iterationen können die übrig gebliebenen Prototypen weiter verfeinert werden.
Im Softwareentwurf kommt hier insbesondere der GUI-Prototyp zum Tragen. Hier sollen durch einfach zu erstellende und auch wieder zu verwerfende Mock-ups Ideen dargestellt werden, die der Benutzer schnell versteht und mit deren Hilfe er rasch die Einsatzfähigkeit und den Nutzen beurteilen kann. Lösungen werden schnell und unkompliziert zu Papier gebracht.

Wie lässt sich Design Thinking integrieren? Drei Varianten

Agile Vorgehensmodelle stellen eine effiziente Arbeitsweise für die Softwareentwicklung dar, nach denen ein Entwicklungsteam Software iterativ und inkrementell entwickeln kann. Entsprechende agile Rahmenwerke, wie beispielsweise Scrum, sind auf die schnelle Entwicklung dessen ausgelegt, was der Kunde zur systemseitigen Lösung seiner Fragestellung benötigt. Aber was ist, wenn wir wirklich innovativ sein wollen?
Der Fokus von Scrum liegt auf der effizi-

enten Entwicklung und der korrekten Umsetzung von Backlog-Einträgen. Das Ziel ist nicht, neue Visionen für ein Projekt zu schaffen. Der Fokus von Design Thinking liegt im Gegensatz dazu auf einer möglichst effektiven, möglichst innovativen Lösung – die Implementierung jedoch gehört nicht dazu, zumindest nicht weiter als ein Prototyp. Das Potenzial liegt somit in der Verbindung beider Ansätze:

- Design Thinking für innovative und effektive vom Nutzer gewünschte Lösungsansätze,
- Scrum für die effiziente und korrekte zeitnahe Umsetzung in Absprache mit dem Kunden.

Der Design-Thinking-Ansatz soll dazu dienen, die Probleme des Anwenders zu verstehen, zu analysieren und durch gute Ideen zu lösen – und nicht dazu, die vermeintlichen Lösungen des Auftraggebers stur umzusetzen. Das Gesamtteam kann mit Hilfe des Design-Thinking-Ansatzes mögliche Lösungen für Probleme sowie deren Ursachen und Beziehungen zueinander aufzeigen, die sowohl dem Kunden als auch dem Nutzer bis dahin noch nicht bewusst waren. Dieses Wissen unterstützt das Projekt und führt zu neuen Lösungsideen.
Eine Integration bedeutet, dass Scrum weiterhin als Vorgehensmodell für ein Entwicklungsprojekt dient und der Design-Thinking-Ansatz dieses zusätzlich unterstützt. Dies kann in unterschiedlichen Varianten erfolgen. Jede der folgenden drei Varianten beeinflusst das Vorgehensmodell auf andere Weise und setzt verschiedene Schwerpunkte bei der Unterstützung. Diese Ansätze beschreiben Prozesse, Aufgabenbereiche, Rollen und Artefakte, die bei einer Integration im Vorgehensmodell Scrum hinzugefügt oder angepasst werden müssen. Die Artefakte der Integrationsansätze unterscheiden sich jedoch in ihrem Inhalt und Umfang im Vergleich zu einem rein nach Scrum durchgeführten agilen Entwicklungsprojekt.
Wir haben verschiedene Möglichkeiten erprobt, die abhängig vom jeweiligen Projektkontext die Paradigmen und die Möglichkeiten des Design Thinking nutzen. Wir unterscheiden insbesondere nach der zeitlichen Einbindung und der Eingliederung in das Scrum-Team.

Variante 1 – Design Thinking vor der Entwicklung

Design Thinking kann vor der Entwicklung, also im klassischen Requirements Engineering, angesetzt werden und so bei der Entwicklung des Product Backlogs helfen. Jede User Story fließt zunächst in ein Pre-Backlog, wo sie durch ein Design-Thinking-Team bearbeitet wird. Hierbei wird der User Story auf den Grund gegangen, ihr zugrundeliegendes Problem hinterfragt, Lösungen verworfen und neu erdacht. Die Entwicklung mündet in einer neuen User Story oder kann auch vollständig verworfen werden. Im Anschluss fließt die User Story in das Product Backlog ein. Von dort durchlebt sie über Sprint Backlog und Entwicklung den bekannten Lebenszyklus. Beim Review kann das Design-Thinking-Team teilnehmen.
Das Design-Thinking-Team unterstützt in Variante 1 den Product Owner bei der Analyse der User Stories. Dieses Team sollte gemäß den Paradigmen des Design Thinking gerade nicht ausschließlich aus Requirements Engineers, fachlichen Architekten und Mitgliedern des Entwicklungsteams zusammengesetzt werden. Das Design-Thinking-Team kann sehr kreativ besetzt werden. Handelt es sich etwa um eine Software zur Unterstützung von Qualitätssicherungsprozessen in der Versicherungsbranche, kann durchaus über einen Qualitätssicherungsberater der städtischen Kindergärten als Teammitglied nachgedacht werden. Ziel ist, eine neue Sicht auf die Endanwender zu und neue Lösungen aus einem anderen Bereich zu erhalten. Auf die Arbeit des Scrum-Teams hat diese Variante keine Auswirkungen.
Variante 1 trägt der klassischen Produktentstehung Rechnung. Hier denken wir typischerweise noch in Releaseplanungen. Mehrere Sprints werden zu einem Gesamtrelease zusammengefasst. Vor jedem größeren Funktionsschub gibt es eine längere Phase des Requirements Engineerings, in der insbesondere die User Stories mit größerem Funktionsumfang zunächst gesammelt werden. Diese werden durch den Product Owner priorisiert.
In diesen Phasen werden die Paradigmen des Design Thinking gelebt und damit dem eigentlichen Problem des Anwenders auf den Grund gegangen, anstatt nur die vorskizzierte Lösung umzusetzen. Dies erfordert einige Zeit des offenen Denkens, der kreativen Analyse des Problems und des Lösungsentwurfs, bis die tatsächlich umzusetzenden User Stories im Backlog landen. Hierbei wird typischerweise die Zeit bis zum nächsten größeren Funktionsschub genutzt. Die Zeitdauer eines einzelnen Sprints ist in der Regel zu kurz, jedoch wird das Design-Thinking-Team an den Reviewterminen der einzelnen Sprints teilnehmen, sodass in Abstimmung mit dem Product Owner nach Rückmeldung der Anwender zum PSI (potentially shipable increment) Anpassungen am Product Backlog vorgenommen werden können. Diese Anpassungen sind jedoch selten so groß wie in der initialen Phase.
Denken wir an Anwendungen, bei denen größere Funktionsumfänge zu einer Auslieferung/einem Release zusammengefasst werden, so werden parallel zur Entwicklung weitere User Stories über die Anwender formuliert, welche nach Einschätzung des Product Owners gegebenenfalls sofort in das Backlog kommen oder durch das Design-Thinking-Team bearbeitet und eventuell erst zur nächsten Auslieferung umgesetzt werden.

Variante 2 – Design Thinking parallel zur Entwicklung

In dieser Variante agiert das Design-Thinking-Team parallel zur Entwicklung direkt auf dem Sprint Backlog des Entwicklungsteams. Das Scrum-Team agiert entsprechend seines Rahmenwerks und entscheidet, wie viele User Stories aus dem Product Backlog umgesetzt werden. Es handelt mit dem Product Owner die Definition of Done aus und setzt die Backlog-Einträge bestmöglich innerhalb des Sprints um. Flankiert wird dies jedoch durch ein Design-Thinking-Team, welches parallel bestehende User Stories überarbeitet, streicht oder neue hinzufügt.
In Sprint Planning, Review und Retroperspektive synchronisieren sich die beiden Teams. Der Product Owner ist Mitglied beider Teams und deren direkte Verbindung zueinander. Das Gesamtteam besteht damit aus einem Scrum-Team und einem Design-Thinking-Team, welche klar voneinander getrennt sind und entsprechend ihrer Rahmenwerke agieren.
Parallel zur Entwicklung und in Zusammenarbeit mit dem Scrum-Team und dem Product Owner entwirft das Design-Thinking-Team Lösungsansätze zur Umsetzung der User Stories, die im nächsten Sprint umgesetzt und anschließend im Review dem Product Owner (und weiteren Anwendern) gezeigt werden. Eine Teilnahme des Design-Thinking-Teams am Review ist daher zwingend erforderlich. Der Scrum Master sorgt dafür, dass beide Teams entsprechend ihrer Aufgabe arbeiten können. Hier ist es erforderlich, dass er mit beiden Rahmenwerken vertraut ist. Er übernimmt die Interaktion zwischen den beiden Teams.

Insbesondere langjährige Softwaregroßprojekte, die Design Thinking intensiv und kontinuierlich in ein Innovationsprojekt integrieren wollen und über langjährige Erfahrung mit Scrum verfügen, profitieren von einem parallelen, aber zeitlich um eine Iteration zum Scrum Sprint nach vorne versetzten Design-Thinking-Zyklus.

Variante 3 – Design Thinking in der Entwicklung

In dieser Variante findet Design Thinking im Scrum-Team statt. In dieser maximal integrierten Kooperationsform wird das Scrum-Team um neue Teammitglieder erweitert. Nun sind nicht mehr nur verschiedene Entwickler, GUI-Designer oder Architekten Teil des Teams, sondern auch weitere interdisziplinäre Rollen. Die Zahl der Teammitglieder sollte sich dennoch am Scrum-Vorschlag von sechs bis acht Mitgliedern orientieren.

Variante 3 ist in ihrer Geisteshaltung dem Design-Thinking-Ansatz am nächsten und kann dazu führen, das Scrum-Gerüst stark aufzuweichen und etwa Sprint-Zyklen nicht mehr fest umzusetzen. Im Umkehrschluss ist das komplette Team in die Lösungsentwicklung eingebunden – inklusive der Entwickler, die nun in der ureigensten Idee des Scrum direkt zur Lösungsidee beitragen.

Das Gesamtteam kann entscheiden, ob es zunächst Prototypen herstellt oder doch ganze Produkte. Ziel ist es, hier Lösungen erst zu entwerfen, zu erproben und dann zu bauen. Das ist nur in einem derart integrierten Team möglich. Die Sprints durchlaufen mindestens einen Design-Thinking-Zyklus. In den Zyklen werden die Ideen entweder direkt gebaut oder erst prototypisch erprobt. Ob direkt nutzbare Software erzeugt wird oder zunächst erst Prototypen, ist auch in anderen Bereichen als dem Softwareengineering eine viel diskutierte Frage. Ein Vorteil ist hier die direkte Einbindung des Teams in die Lösungsentwicklung. Dies führt zu dem Vorteil, die Dokumentation – wenigstens zum Zweck der Softwareausführung – nicht bis ins kleinste Detail ausführen zu müssen, da das Team bereits integriert entwickelt.

Ein Praxisbeispiel – GUI-Redesign

In einem konkreten Projekt standen wir vor der Herausforderung, in einer alten Webanwendung mit sehr viel Fachlogik und formularartigen Masken den Client vollständig neu zu bauen. Ziel des Redesigns war ein intuitives und innovatives Design, das langfristig den Anforderungen der Stakeholder an das System gerecht wird und die darunter liegenden Geschäftsprozesse unterstützt.

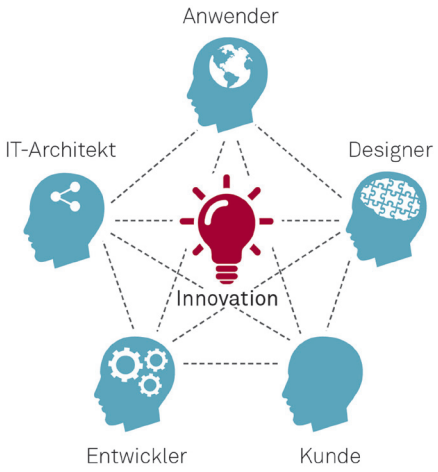

Abb. 3: Das Zusammenspiel des interdisziplinären Teams.

GUI-Design ist der zugänglichste und offensichtlichste Weg, den tatsächlichen Systemanwender mit seinen Wünschen und Vorstellungen einzubinden. Auch können hier sehr einfach Prototypen (schon auf dem Papier) entstehen, mit dem Kunden diskutiert und wieder verworfen werden. So war klar, dass wir für den Umbau eine Methodik verwenden würden, die eine enge Kooperation mit dem Endanwender nutzt.

Der Fachbereich des Kunden hatte bereits die alte Anwendung mitentwickelt und so waren wir auf der Suche nach mehr Inspiration und Möglichkeiten, uns aus den jahrelang verfestigten Denkmustern (die immer wieder zu altbekannten Ergebnissen führten) zu lösen. Erstmalig sollte nicht nur der Endanwender stärker eingebunden werden – es sollten sich auch Experten aus anderen Disziplinen am Redesign beteiligen und ein interdisziplinäres Team formen (**siehe Abbildung 3**).

Auf der Suche nach einem methodischen Rahmen, der uns eben jene Möglichkeiten des offenen Denkens und der Einbindung des Benutzers geben sollte, fanden wir Design Thinking. Das Design-Thinking-Team setzten wir zusammen aus Fachexperten, Anwendern, GUI-Designern und Studenten unterschiedlicher Fachrichtungen.

Wir entschieden uns für ein vorgelagertes Design-Thinking-Team (Variante 1). Aufgrund zeitlichen Drucks standen die User Stories zur Umsetzung sehr schnell bereit, jedoch zunächst mit Inhalten nach altem Muster. Da dies dem Kunden „zu kurz gesprungen“ erschien, lebten wir Variante 2, um parallel zur bereits startenden Entwicklung den Design-Thinking-Ansatz im Projekt weiterzuführen.

Dies führte immer wieder zu Zeitdruck und damit zu Schwierigkeiten bei der Umsetzung des Ansatzes. Wurden viele gute Ideen entworfen, so scheiterten diese häufig nicht an der prinzipiellen Machbarkeit, sondern vor allem an der mangelnden Zeit – der nächste Sprint nahte, es war an der Zeit, eine Lösung auszuwählen (Konvergenz). Doch am Ende von etwa neun Monaten waren stets neue und völlig überraschende Lösungen entstanden, von denen wir viele umsetzen konnten.

Die Einbindung des Endanwenders war nicht immer unkritisch. Zwar kamen viele sehr gute Ideen, es wurden aber auch einige Entscheidungen später erneut hinterfragt. Design Thinking lässt das ausdrücklich zu. Im Implementierungsprojekt, welches mit

Literatur & Links

[Bro06] W. Brody, U.S. Competitiveness: the innovation challenge, siehe: `gpo.gov/fdsys/pkg/CHRG-109hhrg22550/pdf/CHRG-109hhrg22550.pdf`

[Bro09] T. Brown, Change by design. How design thinking transforms organisations and inspires innovation, HarperBusiness, 2009

[Dör09] N. Dörr, T. Müller-Prothmann, Innovationsmanagement. Strategien, Methoden und Werkzeuge für systematische Innovationsprozesse, Carl Hanser Verlag, 2009

[Epp12] M. Eppler, F. Hoffmann, Design Thinking im Management, siehe: `zoe-online.org/content/default.aspx?_s=470591`

[Grot09] A. Grots, M. Pratschke, Design Thinking - Kreativität als Methode, in: Marketing Review St. Gallen, April 2009

[HPI16] Hasso Plattner Institut of Design, HPI Academy, Education for Professionals, `hpi-academy.de`

[Pla11] H. Plattner, Ch. Meinel, L. Leifer (Hrsg.), Design thinking, Springer, 2011

seinen Auslieferungen bestimmte Prozessschritte beim Kunden erreichen muss, ist dies jedoch nicht immer leicht. Dennoch: Waren die Führungskräfte der anwendenden Abteilungen zunächst skeptisch und stellten Mitarbeiter ungern für Interviews und Begutachtungen zur Verfügung, so war nach den ersten entwickelten Prototypen das Feuer schnell entfacht. Wir wurden mutiger und sprachen von einer innovativen Benutzerführung mit Freude am System und dessen Bedienung. Die Fachbereiche reagierten mit stärkerer Einbindung und viel guter Rückmeldung – eine Engelsspirale, die zwar anstrengend war, aber zu guten Ergebnissen führte.

Fazit und Ausblick

Im Redesign der GUI-Oberfläche wurden zwei Varianten des Design Thinkings und drei der Kernparadigmen umgesetzt. Das interdisziplinäre Team sorgte für Diversität, neue Ansätze und ungewohnte Fragen. Die Ideenfülle konnte in Divergenz und Konvergenz erfolgreich in Lösungsansätze verwandelt werden und das offene Denken war dafür verantwortlich, dass der Kunde nicht nur besser eingebunden werden konnte, sondern sich für die Lösungsideen begeistern ließ.

Schmerzlich bewusst wurde uns, dass in der zweiten Variante die kurzen Sprintzyklen grundlegende Überlegungen empfindlich einschränken können. So kamen gute Lösungen aus dem parallelen Team, die aber aufgrund der knappen Taktung zugunsten schneller umsetzbarer (und durchsetzbarer) Ideen verworfen wurden.

Die Varianten 2 und 3, Design Thinking parallel zur Entwicklung und Design Thinking in der Entwicklung, benötigen ein enormes Maß an Disziplin, um die Abfolge von Divergenz und Konvergenz einzuhalten.

Kreativ divergente Phasen müssen sehr konsequent von konvergierender Lösungsfindung abgelöst werden. Droht ein leeres Backlog oder eine später stattfindende Implementierung, ist vieler Kreativität bereits der Raum genommen.

Dennoch entstanden durch beide Varianten neue, überraschende und gut umsetzbare Lösungen. Der nutzerzentrierte Design-Thinking-Ansatz ermöglichte uns, den Kunden aus seinem bisherigen Denken herauszunehmen und die Menge der heterogenen Endanwender zu überzeugen. Aus unübersichtlichen, verschachtelten und überladenen Eingabemasken wurden prägnante, kontextabhängige und intuitive Dialoge. ||

Die Autoren

|| Leif Meyer
(leif.meyer@msg-systems.com)
ist Business Consultant bei msg und beschäftigt sich neben fachlichen IT-Architekturen und agilen Vorgehensmodellen mit methodenbasierter Innovations- und nutzerzentrierter Design-Entwicklung sowie Themen im Bereich User Experience.

|| Dr. Nicole Ondrusch
(nicole.ondrusch@msg-systems.com)
ist Senior Business Consultant bei msg und interessiert sich in ihrer Arbeit im Software-Engineering bei msg für die Verwendbarkeit von neuen Ideen und Entwicklungen im Projektkontext.