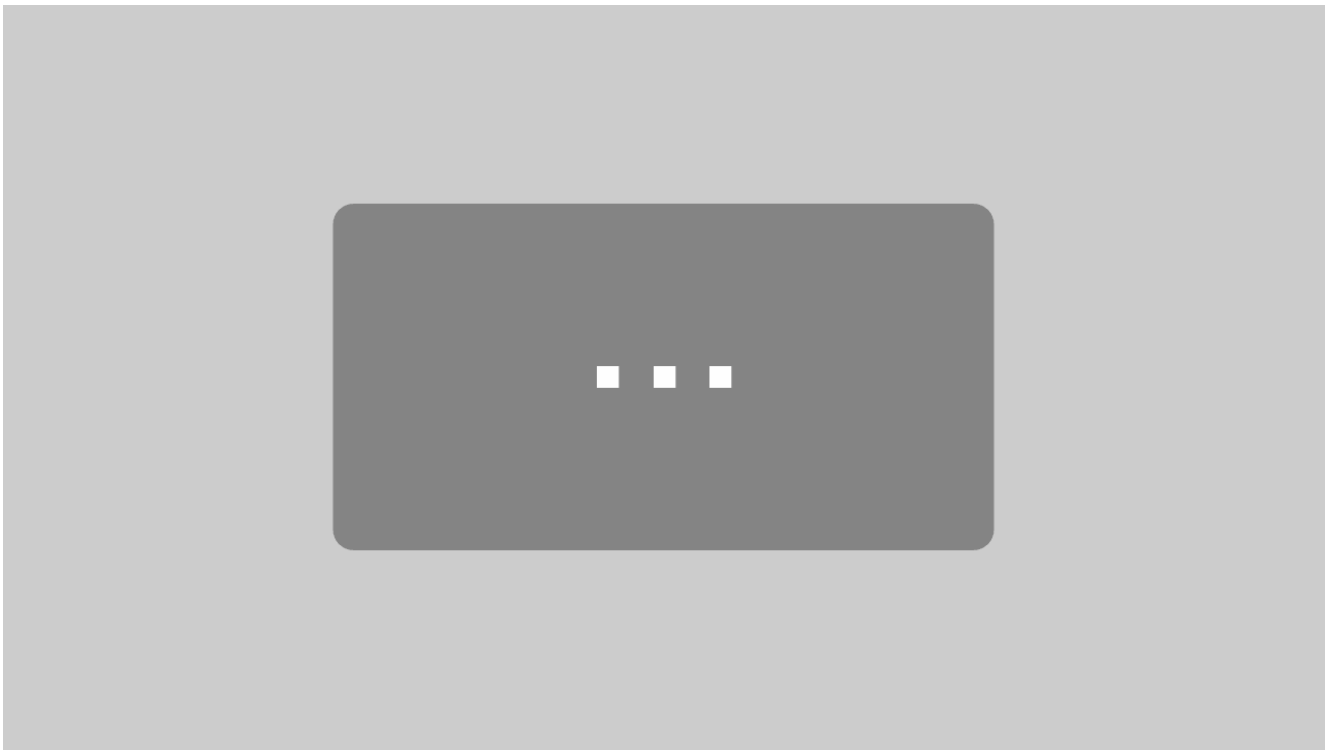
[No 5] Was sagt eigentlich unser Management?

In den Beiträgen **[No 1] –** Alleine ist man weniger zusammen bis **[No 4]** – Auf der Suche nach dem perfekten Termin haben wir uns als IT eCommerce geäußert, wie wir mit der Corona Situation umgehen.

Auf dem Future Retail Forum hat unser Aufsichtsratsvorsitzender Dr. Leif E. Göritz ein Einblick gegeben, wie wir uns als Gesamtunternehmen aufstellen:



Mit dem Laden des Videos akzeptieren Sie die Datenschutzerklärung von YouTube.
Mehr erfahren

Video laden

☐ YouTube immer entsperren

Both sides of the story – Thalia in Zeiten von Corona

Wie schnell sind eigentlich eCommerce Webseiten?

Bei einer Webseite kommt es auf das Nutzungserlebnis / die User Experience an. Es sollen die richtigen Produkte, idealerweise mit schönen großen Bildern angezeigt werden, die Seite soll personalisiert sein, aber die Webseite muss auch schnell laden. Diesen Balanceakt zwischen guten Inhalten und einer schnellen Seite sorgt dafür, dass das Stöbern auf der Seite Spaß macht und das der Besucher idealerweise zu einem Kunden wird. Nur wie messe ich eigentlich „Ladezeit"?

Für das Messen von „Ladezeiten" gibt es zwei Möglichkeiten:

Synthetische Ladezeiten Messung:
Bei der *synthetischen Messung* wird die Webseite in einer „Labor Umgebung" gemessen. Die Seite wird immer vom **gleichen Device**, mit der **gleichen Internetleitung** in einem **festen Browser** getestet.
Da der Client stabil ist, ist es klar, dass es an der Webseite liegt, wenn sich die Ladezeit verändert.
Bei der Auswahl des Clients / der Testumgebungen muss versucht werden **tatsächlichen Besucher** der Webseite zu simulieren (Verbindung, Gerät etc.) da ansonsten die Aussagekraft nur begrenzt ist. Mit dieser synthetischen Messung kann auch jede **beliebige** Webseite gemessen werden und damit ein Benchmark erstellen werden.

Real User Monitoring (RUM):
Um die **tatsächliche** Ladezeit der Besucher zu messen, wird ein *Real User Monitoring* benötigt. Damit wird, innerhalb des Browsers des Besuchers, gemessen wie die Ladezeit der Webseite ist.
Die Varianz ist größer, da die Besucher mit mehr Browsern, Devices etc. reinkommen als in einer Laborumgebung aufbaut werden kann.
Für das Messen von Real User Monitoring gibt es viele Tools. Am verbreitetsten sind Google Analytics, New Relic und boomerang.js.

Im Vergleich zur synthetischen Messung muss in der Regel ein **zusätzliches** JavaScript ausgeführt werden, welches dann einen Impact auf den Besucher hat.
Die Zahlen im Vergleich zu anderen Webseiten zu setzen, ist schwierig, damit ist die Aussagekraft der Messung schwierig.
Wenn jetzt 75% der Besucher einen *First contentful paint* (FCP) von unter 1,5 Sekunden haben, ist das für einen Webshop gut oder schlecht?

Das ist die Aufgabenstellung: Auf Basis von Real User Monitoring ein Marktvergleich von eCommerce Händlern in Deutschland. Mit was für einer **Toolchain** können solche **Daten erhoben und visualisiert** werden?

Chrome User Experience Report (CrUX)

Im Januar 2020 hat der Chrome ein Marktanteil von ca 44%. Sollte der Nutzer nicht widersprochen haben Nutzer Statistiken zu erheben, werden diese Daten automatisch an Google übermittelt und gespeichert. Google stellt diese Daten anonymisiert in seinem*Chrome User Experience Report (CrUX)*zur Verfügung.
Dafür werden die Daten in BigQuery (Google Clouds Data Warehouse) gespeichert und **öffentlich** verfügbar gemacht.

Dieser Report wird monatlich erstellt und beinhaltet eine Reihe an Performance Daten.

Was kann damit ausgewertet werden?

Mithilfe von BigQuery können historische Daten (ab Oktober 2017) auf Domainsbasis ausgewertet werden.
Die KPIs sind sowohl Endgeräte-spezifisch (Mobil, Desktop) als auch Verbindungs-spezifisch (4G, 3G etc.)

```
SELECT
    SUM(bin.density) AS density
FROM
    `chrome-ux-report.country_de.202001`,
    UNNEST(first_contentful_paint.histogram.bin) AS bin
WHERE
    bin.end <= 1500 AND
    origin = 'https://www.thalia.de'
```

```
    AND form_factor.name = "phone"
ORDER BY
    density DESC

-- Ergebnis = 61%
```

Dieser Abfrage sagt aus, dass 61% der deutschen Mobilen Chrome User ein First Contentful Paint von <= 1,5 sec bei thalia.de haben. Die Query lässt sich dann auch leicht abwandeln, um Informationen für andere Domains oder auch Ländern zu bekommen.

Damit können auf **Monatsbasis** die Werte ermittelt werden.

Als Alternative zu BigQuery stehen auch fertige Lösungen wie https://crux.run zur Verfügung.

Es gibt allerdings noch zwei Themen, die etwas stören:

1. Daten werden auf Monatsbasis aggregiert. Monatlich eine „Überraschung“ zu bekommen ist doof. Idealerweise soll auf **tägliche oder wöchentliche Basis** ein Trend ermitteln werden können.
2. Es werden nur aggregierte Daten **für eine Domain** angezeigt nicht für **einzelne Seiten**.
   Aber unterschiedliche Seitentypen haben unterschiedliche Ladezeiten. Eine Startseite hat andere Herausforderungen als eine Artikeldetailseite. Idealerweise werden einzelne Seitentypen gemessen und diese miteinander verglichen.

PageSpeed Insights (PSI) API für detailliertere RUM Messungen

PageSpeed Insights ist eine Toolsuite, welches mehrere Funktionalitäten in einer Oberfläche anbietet.

- Lighthouse als synthetisches Messtool, welches KPIs ermittelt und auf Best Practices überprüft. Das Bekannteste ist der Lighthouse Score. Ein Wert, der aussagt in wie weit Best Practices eingehalten werden.
  Lighthouse ist ansonsten auch innerhalb von Chrome & Firefox enthalten.
- **Tagesaktuelle Crux Daten** PSI stellt tagesaktuelle Crux Daten zur

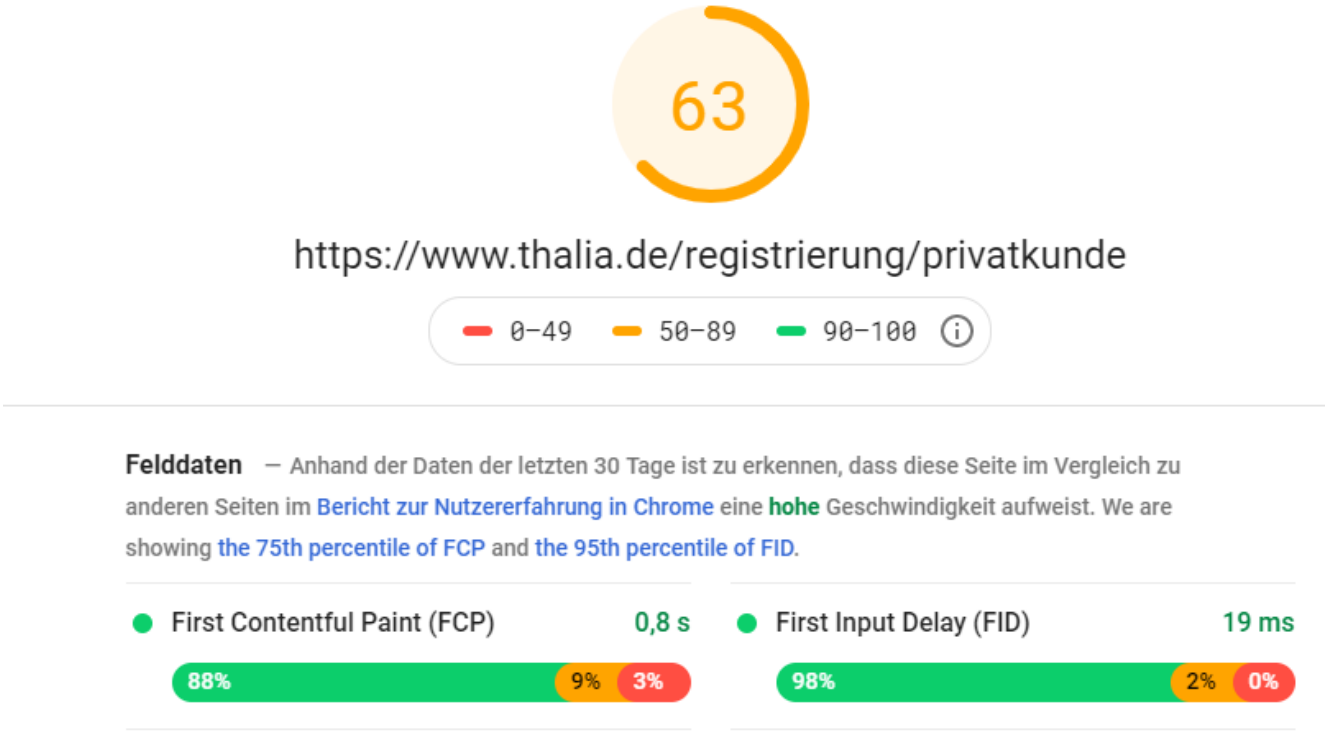
Verfügung. Die Daten werden täglich aktualisiert, beinhalten aber den **Durchschnitt aus den letzten 30 Tagen rollierend**.

- **Ausgewählte Crux Daten für einzelne Seiten** im BigQuery und damit öffentlich verfügbar, sind die Daten für eine komplette Domain. Innerhalb von PSI werden **zusätzlich** auch noch die Daten für einzelne Seiten dargestellt.
  Bedingung ist, dass die Seite genug Traffic hat und somit eine anonymisierte Auswertung möglich ist.
- PSI stellt nur ausgewählte Daten zur Verfügung.
  PSI fokussiert sich auf *First Contentful Paint* (FCP) und *First Input Delay (FID)* und stellt nur diese Daten zur Verfügung.
  Weitere KPIs (TTFB, DCL, etc.) werden aktuell **nicht angezeigt**
- **Eine Bewertung der KPIs**: PSI gruppiert die KPI Werte in drei Bereiche *schnell, durchschnittlich, langsam.* Dabei gelten aktuell folgende Grenzwerte, unabhängig vom Gerätetyp:

	Schnell	**Durchschnittlich**	**Langsam**
FCP	[0–1.000 ms]	[1.000–2.500 ms]	über 2.500 ms
FID	[0–50 ms]	[50–250 ms]	über 250 ms

- Zusätzlich gibt PSI für den FCP den **75% Quantil** und bei FID den **95% Quantil** zurück und stuft die Seite nach diesem Wert in schnell, durchschnittlich oder langsam ein.

Lesebeispiel: Wenn die Registrierungsseite für Privatkunden in PSI bewertet wird,
…: https://developers.google.com/speed/pagespeed/insights/?hl=de&url=https%3A%2F%2Fwww.thalia.de%2Fregistrierung%2Fprivatkunde&tab=mobile



… dann hat die Seite auf **mobilen Endgeräten:**

- Durchschnittliche (63 Punkte) Lighthouse Bewertung.
- 88% der Besucher eine schnelle FCP Zeit hatten (< 1.000 ms)
  und das der 75% Quantil bei 800 ms liegt.
  Leider haben 3% eine FCP von über 2,5 Sekunden.
- 98% eine schnelle FID (<50ms) hatten und das kein FID über 250ms liegt.
- die Seite sowohl in FCP als auch FID „schnell“ ist (grüner Punkt) und damit die Seite insgesamt „schnell“ ist.

Damit jetzt nicht jedes Mal die Webseite aufgerufen werden muss, stellt PSI stellt auch eine kostenfreie API zur Verfügung.

Monitoring & Dashboard Lösung

Was soll erreicht werden?
Ein Report der die PCI & FID für unterschiedliche Domains anzeigt.

Was muss dafür gemacht werden?

- die Rest API aufrufen
- Daten speichern
- täglich / wöchentlich aktualisieren

- Daten visualisieren
- für den Fehlerfall ein Monitoring / Jobsteuerung aufsetzen

Von der Datenmenge her ist es eine CSV Datei, die auch in Excel / Spreadsheet gespeichert werden kann.
Warum nicht also per Google Spreadsheet realisieren?

AppScript

Innerhalb von Google Spreadsheet gibt es eine Chrome V8 Laufzeitumgebung, die sich Apps Script nennt. Scripte können veröffentlicht werden und es existiert ein Ökosystem für Entwicklung, Jobsteuerung / Trigger, Monitoring, Fehlermails etc.
Es gibt Limitierungen, so kann z.B. weder asynchron gearbeitet werden, noch gibt es Quotas, die je nach GSuite modell unterschieden werden. Für diesen Benchmark reicht jedoch die kostenfreie Version aus.

Mit folgendem Code Snippet wird PSI aufgerufen und in einer Tabelle gespeichert.

```
function callPageSpeedV5(url, strategy) {
                    var          pageSpeedUrl          =
'https://www.googleapis.com/pagespeedonline/v5/runPagespeed?ur
l=' + encodeURIComponent(url) + '&key=' + pageSpeedApiKey +
'&strategy=' + strategy;
  console.info(pageSpeedUrl);
  try{
    var response = UrlFetchApp.fetch(pageSpeedUrl);
    var json = response.getContentText();

    return JSON.parse(json);
    }catch (e) {
        console.error('callPageSpeedV5 Error on ' + url + ":"
+ e);
    }
}


function writeJSONtoSheet(result, strategie) {
                          var            sheet            =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("data");
    sheet.appendRow([Utilities.formatDate(new Date(), 'GMT',
```

```
'yyyy-MM-dd'),
                    result.id,
                    "FCP",
result.metrics.FIRST_CONTENTFUL_PAINT_MS.percentile,
result.metrics.FIRST_CONTENTFUL_PAINT_MS.distributions[0].prop
ortion,
result.metrics.FIRST_CONTENTFUL_PAINT_MS.distributions[1].prop
ortion,
result.metrics.FIRST_CONTENTFUL_PAINT_MS.distributions[2].prop
ortion,
result.metrics.FIRST_CONTENTFUL_PAINT_MS.category,
result.metrics.FIRST_INPUT_DELAY_MS.percentile,
result.metrics.FIRST_INPUT_DELAY_MS.distributions[0].proportio
n,
result.metrics.FIRST_INPUT_DELAY_MS.distributions[1].proportio
n,
result.metrics.FIRST_INPUT_DELAY_MS.distributions[2].proportio
n,
result.metrics.FIRST_INPUT_DELAY_MS.category,
                    strategie
   ]);


}
```

Datastudio

Bisher ist die Datenablage und die Datenaktualisierung geregelt. Jetzt müssen die Daten noch visualisiert und verfügbar gemacht werden.
Eine Lösung hierfür ist Data Studio. Data Studio ist ein Cloud Tool, mit dem unterschiedliche Datenquellen angebunden und visualisiert werden können. Es gibt Daten Connectoren sowohl für Services (z.B. Facebook, Google Analytics, Spreadsheets) als auch Technologien (MySQL, PostgreSQL, BigQuery etc.). Insgesamt sind es ca. 200 Daten Connectoren, die sowohl kostenlos, kommerziell oder open Source sein können. Wenn der gewünschte Daten Connector nicht vorhanden ist, kann auch ein eigener entwickelt werden.

Damit existiert eine Möglichkeit mit der die Daten automatisch aktualisiert und visualisiert werden.

eCommerce Website Performance Dashboard

Im Report sind ca. 130 eCommerce Händler Vertreten. Für diese Händler wird auf täglicher Basis die PSI Daten ermittelt und visualisiert. Dabei wird sowohl nach Device unterschieden und es werden die Daten für die Startseite und die komplette Domain ermittelt.
Glückwunsch gehen nach Ibbenbüren wo https://www.musik-produktiv.de die s*tand heute*, **einzige** Domain haben, die Mobil sowohl in FCP als auch FID „fast“ ist.

Wir bauen unseren Webshop um

Eine Frage, die wir uns schon lange gestellt haben, war: was wollen wir mit unserem Webshop machen? Dieser ist mittlerweile etwas in die Jahre gekommen. Wir sind zum Entschluss gekommen, er **passt nicht mehr,** und wir wollen ihn ablösen.

Fragen wie

- Wie leicht können Änderungen durchgeführt werden?
- Wie gut passt er zu unserem fachlichen Zielbild?
- Wie gut passt er zu unserer Organisation (Conway's Law)?

sind für uns wichtige Kriterien und haben zu dieser Entscheidung geführt.

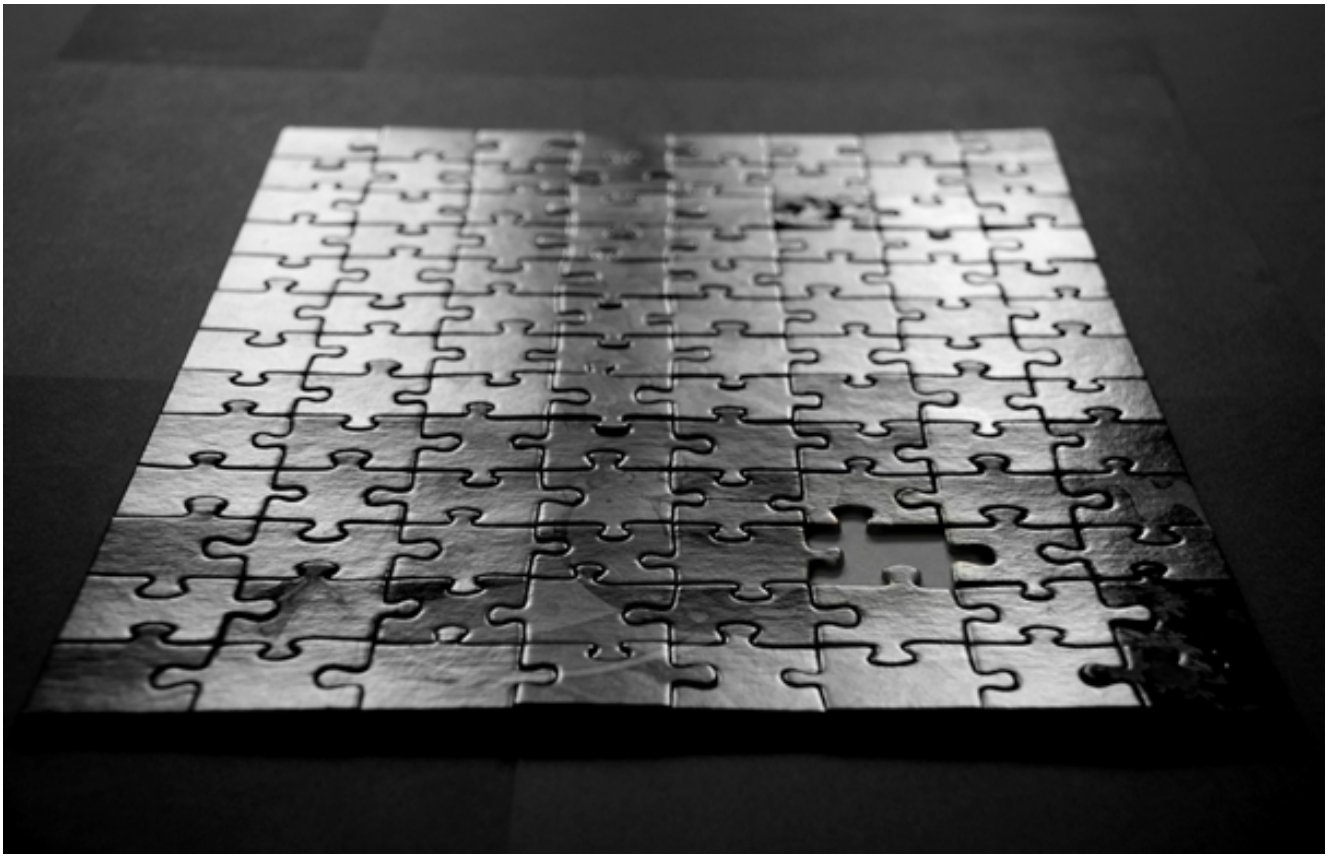
Nur durch was wollen wir unseren Webshop ersetzen?

Wenn wir etwas ablösen wollen, müssen wir auch sagen, wodurch wir das ablösen wollen. Die erste Frage ist: wie groß ist eigentlich das „Greenfield"?

Neben den gewünschten Funktionalitäten gibt es noch eine Reihe weiterer Kriterien zu beachten:

- Wie gut integriert es sich in unsere bestehende IT Landschaft?
  Wir wollen nicht unsere komplette IT Landschaft (ERP, Suche, CRM, Analytics,…) umstellen. Nach dem „best of breed"-Ansatz sind da zum Teil Kauflösungen unterschiedlicher Hersteller zum Teil Eigenentwicklungen dabei.
- Wie integriert es sich mit der bestehenden Organisation & Prozesse?
  Mit z.B. .NET kennt sich hier kaum jemand aus ;-)), wir haben cross-funktionale Produktteams, die das realisieren sollen.
- Wie weit passt es zu unserem geplanten Projektverlauf?
  Wir wollen iterativ vorgehen und parallel weitere Projekte realisieren.

Auswahl des Shopsystems



Tim Geers you make it complete CC BY 2.0

Es gibt ziemlich viele eCommerce Systeme – Open Source, On Premise, SaaS Lösungen etc. Trotzdem haben wir uns gegen eine Standardlösung entschieden und realisieren hauptsächlich mit einer Individuallösung. Wir wollen Systeme haben die sich **uns** anpassen und nicht unsere Organisation, Prozesse, fachliche Features **an Systeme** anpassen. Ein Standardsystem zu nehmen und so **extrem** anzupassen, dass es unseren Wünschen entspricht, halten wir für keine gute Idee.

Zielarchitektur definieren

Welche Architektur ist für uns die richtige? Wir sind von der Organisation und Komplexität so groß, dass wir uns in mehrere Teams aufteilen müssen. Daher müssen wir in der Makroarchitektur einen **Domänenschnitt** finden, der den besten Trade Off zwischen möglichst kleinen Einheiten, möglichst wenigen Schnittstellen, möglichst hoher Kohäsion, möglichst hoher Änderbarkeit, möglichst stabilem Schnitt, möglichst geringer Kosten und noch vieles mehr bildet.

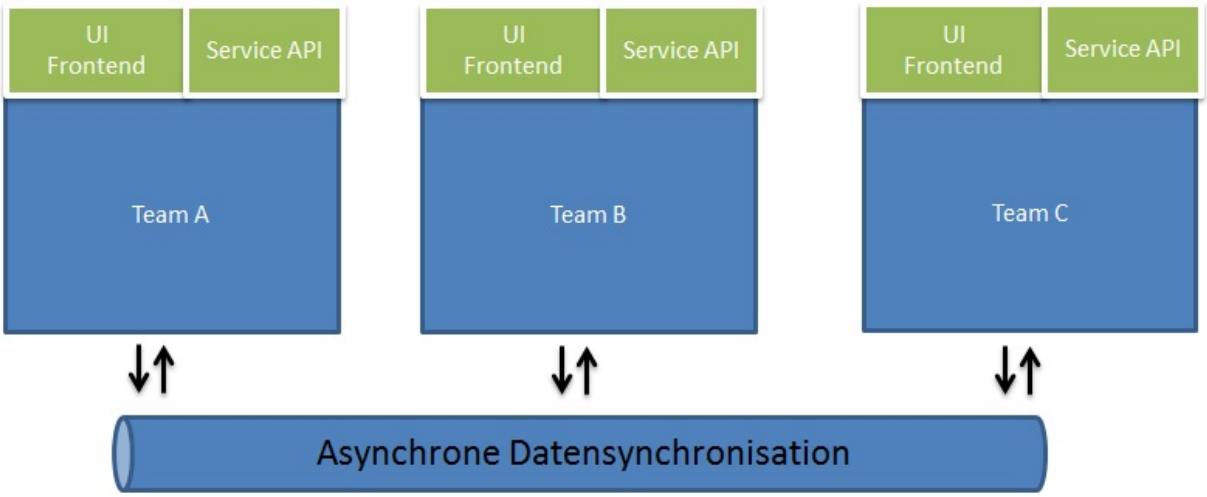
Domänenschnitt

Wir haben uns dafür entschieden, dass eine Kundenfunktionalität möglichst komplett von einem einzigen Team betreut wird. D.h., ein Team ist z.B. für die Artikelsuche zuständig – und das idealerweise von der Visualisierung zum Kunden über die eigentliche Suche bis zur Datenversorgung der Suchmaschine. So sind unsere Produktteams geschnitten und so sollen sie sich auch in unserer technischen Abbildung widerspiegeln.
Da wir nicht nur eine Webseite haben sondern auch Native Apps und spezielle eReader Anwendungen, sind unsere Frontend Technologien sehr vielfältig.

Daher haben wir uns zu einem **Kompromiss** entschieden. Für Webseiten geht die Teamverantwortung bis in die Visualisierung. Für Native Anwendungen greifen unsere Frontend Spezialisten auf Service APIs zu. Er ist also ein Funktionaler Schnitt für Webseiten und eine Schichtenarchitektur für Native Frontends.
Wir orientieren uns für die Webseiten an der SCS Architektur, allerdings haben wir viele Service APIs und stellen zum Teil beide APIs zur Verfügung – eine HTML

Webseite und eine native Suche, die dann im Hintergrund eine Rest-API verwendet. Wir haben uns dafür entschlossen, da Webseiten unser Hauptfokus sind und wir nicht immer einer Service API brauchen. Wie immer gibt es Vor- und Nachteile, aber diese Lösung fühlt sich gut an und hilft uns, unsere Omnichannel Plattform aufzubauen.

Innerhalb der Teamgrenzen ist es jedem Team freigestellt, wie sie ihre Domäne aufteilen. In der Praxis zeigt sich, dass diese in mehreren Microservices aufgeteilt wird – zum Teil auch hier ein Funktionaler Schnitt zum Teil nach Schichten. Wie das in der Praxis aussieht, kann man z.B. bei Team Kaufen sehen.



Kommunikation zwischen den SCS Systemen

Für die Kommunikation zwischen den SCS Systemen setzen wir bei synchroner Kommunikation auf REST Schnittstellen und bei asynchroner Kommunikation auf Events. Die Events werden über ein Messaging System verarbeitet und verteilt. Wir wollen möglichst asynchron kommunizieren. Wir klären gerade, wie wir allen Systemen diese Technologien beibringen. Zum Teil müssen wir noch Lösungen finden, da nicht alle bestehende Standardsysteme solche Schnittstellen anbieten.

Zusammenfassung

Wir wollen einen Webshop haben, welcher zu unserer neuen Organisation passt und sich leicht verändern lässt. Da unser jetziger Webshop dies nicht ausreichend leistet, wollen wir in ablösen.
Eine Standardlösung passt nicht, daher haben wir uns entschlossen, eine Individuallösung zu realisieren, die im Kern auf eine SCS Architektur aufsetzt. Für die nativen Frontends werden Service APIs verwendet.

Was machen wir mit unserem „Webshop?“

Unser Webshop

Start der Entwicklung für unseren Webshop war im Jahre 2004. Zu der Zeit waren wir (damals noch buch.de internetstores AG) noch ein reines eCommerce Unternehmen, und mit diesem Fokus wurde unser Webshop entwickelt.

Thalia.de 2004
Webseite so sah die
aus

Seitdem ist viel passiert…

Unser Business Model und unser Unternehmen haben sich verändert

- Ebooks sind stark gewachsen und die Tolino Allianz hat sich gegründet
- Schon 2004 hatten wir Multichannel-Funktionalitäten (z.B. Filialabhohlung), aber spätestens mit dem Zusammenschluss von buch.de & Thalia lag der Fokus auf „Cross-Channel“, und die Integration der unterschiedlichen Kanäle hatte begonnen.

Jetzt befinden wir uns auf dem Weg zum Omnichannel-Unternehmen und versuchen, die tatsächliche Verschmelzung der Kanäle zu realisieren. Die ersten Ansätze sind da, aber die Frage ist natürlich, wie weit können wir diese Reise mit unserem aktuellen Webshop gehen?

Die Methode, mit der wir Software entwickeln, hat sich verändert

Ähnlich wie sich das ganze Thema „Agile Softwareentwicklung“ entwickelt hat, haben wir uns natürlich auch verändert und setzten aktuell auf Scrum in einer Produktorganisation.
Da Conway’s Law eine große Rolle spielt, ist die Frage: Inwieweit passt unser Webshop in unsere neue Produktorganisation?

Die Technologien und Architekturen, mit denen wir Software entwickeln, haben sich verändert

Glücklicherweise wechselten wir 2004 auf das Spring Framework, und Spring wird immer noch aktiv weiterentwickelt – keine Selbstverständlichkeit, da ja Spring ebenfalls erst in 2004 in der Version 1.0 veröffentlicht wurde. Haben wir

evtl. eine der ältesten noch aktiv weiterentwickelten Spring Anwendungen ? Natürlich haben wir unseren Webshop regelmäßig weiterentwickelt und sind nicht mehr auf Spring 1.0, aber ein Kern und die ursprünglichen Prinzipien blieben bestehen.
Auf der anderen Seite ist die „Best-in-Suite vs Best-in-Breed“ Debatte. Es gibt neuerdings immer mehr spezialisierte Tools / Frameworks, die in ihren Spezialgebieten besser sind (NoSQL Datenbanken, JavaScript Frameworks, alternative JVM Sprachen). Die Frage ist: Inwieweit können wir unseren Webshop mit den unterschiedlichen Tools kombinieren? Reichen uns die Möglichkeiten, die wir haben, aus oder müssen wir feststellen, dass sich die zukünftigen Business-Anforderungen nur mit Technologien umsetzen lassen, die wir nicht einsetzen können?