Matthias Reining

»Step by Tab
Fehler- und Abbruch-Tracking bei formularbasierten Webprozessen

Zu lange und zu komplizierte Online-Formulare schrecken viele User ab. Wer im Internet etwas verkaufen möchte, sollte daher gerade bei der Abfrage von Benutzerdaten großen Wert auf Usability legen. Aber an welchen Stellen im Prozess bleiben die User genau hängen? Und aus welchen Gründen geben potenzielle Kunden vorzeitig auf? Mit dem Tracking von Fehlermeldungen und Abbruchgründen können Unternehmen mögliche Problemfelder von Webformularen identifizieren – und gewinnen so wichtige Informationen zur Steigerung der Konversionsrate.

Um heute ein Online-Geschäft abzuschließen, ist es in der Regel zunächst notwendig, eine Vielzahl persönlicher Daten preiszugeben. Beim Online-Shopping, bei der Nutzung von Finanzdienstleistungen, aber auch bei der Anmeldung für ein Partnervermittlungs-Portal kommen Benutzer um das Ausfüllen langer Webformulare nicht herum. Die erforderlichen Daten werden dabei meist in mehreren aufeinanderfolgenden Schritten erfasst, da die Eingabemasken für eine einzelne Webseite zu umfangreich sind. So müssen Kunden beispielsweise für den Abschluss einer Kfz-Versicherung oft mehr als 50 unterschiedliche Formularfelder ausfüllen – von den persönlichen Informationen des Versicherungsnehmers über die Fahrzeugdaten bis hin zu den Tarif- und Einstufungsdetails.

Viele User sind damit überfordert. In Studien wurde nachgewiesen, dass die Abbruchquoten bei bis zu 75 Prozent liegen, wenn Formulare nicht benutzerfreundlich gestaltet sind. Unklare Formulierungen oder zu komplexe Abfragen führen dazu, dass potenzielle Kunden ihre Bestellung nicht abschließen und stattdessen zum Webangebot des Konkurrenten wechseln. Mittlerweile haben viele Unternehmen dieses Problem erkannt und befassen sich intensiver mit der Usability ihrer Webformulare. Die möglichen Optimierungsansätze sind vielfältig: Es kann zum Beispiel sinnvoll sein, die Reihenfolge der Formularfelder zu ändern, Freitextfelder durch Drop-Down-Menüs zu ersetzen oder automatische Vorbelegungen mit den tatsächlich möglichen Auswahloptionen einzurichten. Hinweistexte sollten dem Benutzer erklären, warum er bestimmte Felder ausfüllen muss und welche Auswirkungen seine Eingaben haben. Und in manchen Fällen empfiehlt es sich schließlich, optionale Abfragen einfach wegzulassen.

Welche Optimierungsmaßnahmen im Einzelfall die größten Auswirkungen auf die Abschlussquote haben, hängt dabei in erster Linie von der jeweiligen Zielgruppe ab. Die Analyse des Nutzerverhaltens spielt daher für formularbasierte Webprozesse eine immer wichtigere Rolle. Unternehmen müssen dazu in der Lage sein, die Art und Häufigkeit der auftretenden Fehlermeldungen zu messen und die Abbruchgründe zu tracken. Im Folgenden wird die technische Umsetzung eines derartigen Tracking-Systems am Beispiel der Web-Controlling-Lösung etracker dargestellt. Die beschriebenen Konzepte lassen sich aber auch mit anderen gängigen Analysetools umsetzen.

Tracking von Fehlermeldungen

Um einen formularbasierten Webprozess stetig zu verbessern, ist es wichtig zu wissen, bei welchen Feldern besonders viele User auf Fehler gestoßen sind. Typischerweise werden alle Eingaben des Benutzers auf dem Server validiert. Dabei kann nicht nur kontrolliert werden, ob sämtliche Pflichtfelder ausgefüllt wur-

DER AUTOR

Matthias Reining ist Softwareentwickler und -architekt bei der nobisCum Deutschland GmbH.

den – es lässt sich zum Beispiel auch automatisch überprüfen, ob die Postleitzahl zum eingegebenen Ort passt oder ob die Bankleitzahl und das ausgewählte Kreditinstitut übereinstimmen.

Schlagen hier eine oder mehrere Validierungen fehl, bekommt der User normalerweise eine Fehlermeldung angezeigt. Das Tracken dieser Fehlermeldungen in aggregierter Form kann Hinweise darauf geben, ob die User immer wieder über die gleichen Eingabefelder stolpern. In diesem Fall sollte das User Interface optimiert werden beispielsweise durch Drop-Down-Menüs, zusätzliche Erläuterungen oder eine automatische Vorbelegung bestimmter Felder. Im Folgenden wird ein Lösungsansatz aufgezeigt, um eine Integration dieser Validierungsfehler in etracker zu realisieren. Im etracker selbst können dann die aggregierten Ergebnisse ausgewertet werden.

Zwar bietet etracker „out of the box“ keine eigenen Funktionen, um Fehler auf Feldebene zu tracken. Der integrierte Event-Tracker kann jedoch für diesen Zweck verwendet werden. Ein Einsatz des Klick-Trackers wäre an dieser Stelle ebenfalls denkbar –diese Variante wird im Abschnitt „Abbruchgründe tracken“ näher erläutert.

Der folgende Beispiel-Code bezieht sich auf das populäre Java-basierte WebFramework Struts 2 (siehe *http://struts.apache.org/2.x/*). Das grundsätzliche Konzept lässt sich aber auch in jeder anderen gängigen Technologie umsetzen.

Validierungsfehler im System registrieren

Fehler können auf dem Backend-System beispielsweise wie folgt registriert werden:

```
public boolean checkFields() {
  ...
  addFieldError("firstname", getText("mandatory"));
  ...
  addFieldError("startdate", getText("notValid"));
  ...
}
```

Fehler an etracker melden

Die ausgelieferte HTML-/JSP-Seite enthält dann meist im oberen Abschnitt einen Informationsblock, der die aufgetretenen Fehler für den User auflistet. Mit diesem Block können die Fehler auch dem Event-Tracker von etracker mitgeteilt werden.

```
<s:if test="hasErrors()">
  <script language="JavaScript">
    <s:iterator var="error" value="getErrorFields4Tracker()">
      ET_Event.eventStart('Fehlermeldung', 'Seitenname',
                          '${error.key}', '${error.value}');
    </s:iterator>
  </script>
  <div class="error-container">
    <s:fielderror />
  </div>
</s:if>
```

In dem Beispiel wird die Methode **getErrorFields4Tracker** aufgerufen. Diese Methode ist auf dem Server implementiert und liefert die Struts-Fehlermeldungen in aufbereiteter Listenform zurück. Die Fehler werden in einer Schleife (Struts2 **s:iterator-Tag**) abgearbeitet. Bei jedem Durchlauf wird die JavaScript-Funktion **ET_Event.eventStart** aufgerufen, die dafür sorgt, dass die Daten an den Event-Tracker geschickt werden.

Dabei werden **Fehlermeldung** und **Seitenname** zur Kategorisierung verwendet. Der Parameter **error.key** enthält den Feldnamen und **error.value** die eigentliche Fehlermeldung.

Abbildung 1: Anzeige der Fehlermeldungen im Event-Tracker von etracker

Abbildung 2: Anzeige der Fehlermeldungen inklusive Fehlergründe im Event-Tracker von etracker

Fehler auswerten

Im etracker kann sich der Web-Administrator die entsprechenden Fehler dann übersichtlich anzeigen lassen.

In Abbildung 1 lässt sich nun erkennen, auf welcher Seite ein Fehler aufgetreten ist (hier: „Workflow-Page1“ bzw. „Workflow-Page2“ in der Spalte „Objekt“). Hinter der Spalte „Aktion“ verbirgt sich das fehlerhafte Feld. Die Zahlen 43, 31 und 25 geben an, wie häufig der entsprechende Fehler aufgetreten ist. Die folgende Spalte zeigt, wie viele unterschiedliche User den Fehler verursacht haben. In diesem Fall steht der Wert jeweils auf „1“, da die Fehler aus einer Websession gezielt für diesen Artikel provoziert wurden.

Die oben dargestellten Fehlerinformationen sind im etracker bei der Standard-Einstellung „Ansicht: ohne Tags – Object + Aktion + Kategorie“ sichtbar. Um sich die unterschiedlichen Fehlermeldungen pro Feld anzeigen zu lassen, muss der Benutzer die Ansicht „mit Tags“ auswählen. Diese Einstellung wird in der blauen Menüleiste rechts oben vorgenommen.

In Abbildung 2 sind nun auch die Fehlergründe eingeblendet. Sie werden hier einfach durch ein Komma getrennt angezeigt. Sollen die Fehlergründe separat getrackt werden, sodass sie individuell auswertbar sind („Wie oft ist exakt dieser Fehler aufgetreten?“), muss der Aktionstyp eindeutig sein (in diesem Beispiel etwa die Servervariable error.key).

Um die Fehlertypen der einzelnen Formularfelder übersichtlicher anzuzeigen, empfiehlt es sich, die Darstellung etwas anzupassen: So kann beispielsweise die Spalte „Objekt“ für den Feldnamen genutzt werden, die Spalte „Aktion“ für den Fehlertyp und die Spalte „Kategorie“ für ein Kombinationsfeld aus Fehlermeldung und der Seite, auf der der Fehler aufgetreten ist.

Tracking von Abbruchgründen

Die Auswertung der Validierungsfehler liefert in der Regel bereits zahlreiche Anhaltspunkte für die Optimierung von Webformularen. Allerdings lässt sich allein auf Basis dieser Informationen noch nicht beurteilen, warum eine bestimmte Anzahl von Benutzern das Ausfüllen des Formulars vorzeitig beendet hat. Um mehr über die Abbruchgründe zu erfahren, sollten insbesondere die letzten Fehler der User betrachtet werden, unmittelbar bevor diese den Workflow-Prozess verlassen haben. Die letzten Fehler müssen zwar nicht immer die Abbruchgründe der User widerspiegeln, sind aber ein guter Indikator für mögliche Probleme, welche die Benutzer am weiteren Prozessfortschritt hindern.

Beim folgenden Ansatz wird daher nur die letzte Fehlermeldung eines Users gesondert erfasst, da es sich hier mit hoher Wahrscheinlichkeit um einen möglichen Abbruchgrund handelt. Diese „Abbruchfehler“ zu ermitteln, ist etwas komplexer als das Tracking der Validierungsfehler, denn es lässt sich nicht ohne Weiteres erkennen, zu welchem Zeitpunkt der User den Prozess abgebrochen hat. Zum einem liegt dies am zustandslosen http-Protokoll, zum anderen an der Tatsache, dass die Webanwendung nicht registriert, wenn der Benutzer die Anwendung oder den Browser einfach schließt. JavaScripts-Events wie onunload oder beforeunload helfen an dieser Stelle nicht weiter: Der Google-Browser Chrome reagiert darauf beispielsweise mit einer unschönen Dialogbox, die darauf hinweist, dass jetzt noch weiterer JavaScript-Code ausgeführt wird – ein No-Go im B2C-Bereich!

Im ersten Schritt gilt es also festzustellen, wann der User den Prozess überhaupt abbricht. Prozessorientierte Webanwendungen verwalten ihre Daten typischerweise in einer Websession. Jede Session ist solange gültig, bis ihr Timeout erreicht wird. Hierbei handelt es sich um eine auf dem Server konfigurierbare Zeitdauer, die bei jeder Client-/Browseranfrage zurückgesetzt wird. Ist das Timeout erreicht, wird die Session auf „invalid“ gesetzt. Bevor dies passiert, triggert die Webanwendung typischerweise für genau diese Websession noch einmal ein Event, in das man sich beim Tracking einhängen kann. Genau in diesem Moment ist sichergestellt, dass der User den Prozess tatsächlich beendet hat.

Die folgenden Code Snippets zeigen, wie dies beispielsweise in einer Java-Enterprise-Anwendung-(JEE) umgesetzt wird:

WEB-INF/web.xml

```
<web-app>
  <!-- ... -->
  <session-config>
    <!-- 10 minutes -->
    <session-timeout>10</session-timeout>
  </session-config>

  <listener>
    <listener-class>
      de.nobiscum.stebbytab.TrackingSession-
Listener
    </listener-class>
  </listener>
  <!-- ... -->
</web-app>
```

In der web.xml wird ein spezieller Listener registriert. Dieser implementiert das Interface HttpSessionListener und wird immer beim Erstellen und Zerstören einer Websession aufgerufen.

Klasse TrackingSessionListener.java

```
package de.nobiscum.stebbytab;

import javax.servlet.http.HttpSession;
import javax.servlet.http.HttpSessionEvent;
import javax.servlet.http.HttpSessionListener;

public class TrackingSessionListener imple-
ments HttpSessionListener {

  @Override
  public void sessionCreated(HttpSessionEvent
se) {
  }

  @Override
  public void sessionDestroyed(HttpSessionE-
vent se) {
    HttpSession session = se.getSession();
    System.out.println("Der Prozess ist zu
Ende oder wurde abgebrochen");
  }
}
```

Nachdem eine Stelle gefunden wurde, bei der man sicher sein kann, dass der Prozess beendet bzw. abgebrochen wurde, kann nun das eigentliche Tracking beginnen.

Prozessabbruch-Fehler mit etracker erfassen

Während das Tracking der Validierungsfehler auf der Client-Seite erfolgte, befinden wir uns nun im Backend der Webanwendung. Da der User seinen Browser bereits geschlossen hat bzw. auf anderen Webseiten surft, können die Fehler nicht mehr mithilfe der normalen JavaScript-Verfahren des etracker erfasst werden.

Als eine weitere Möglichkeit, dem Tracking-System Informationen zukommen zu lassen, bietet etracker den Klick-Tracker. Eigentlich ist dieses Feature dazu gedacht, den Download von Dokumenten zu tracken. Es lässt sich aber auch sehr gut für die Erfassung der Prozessabbruch-Fehler nutzen.

Wenn die Websession des Benutzers zerstört wird, wird geprüft, ob in ihr noch fehlerhafte Felder liegen. Ist dies der Fall, ruft man für jeden Fehler den Klick-Tracker mit einer speziellen Fehlerinformation auf. Allgemein sieht die URL für den Klick-Tracker wie folgt aus:

```
http://www.etracker.de/lnkcnt.php?et=SICHER-
HEITSCODE&url=ZIEL_URL&lnkname=BEZEICHNUNG_IN
_DER_STATISTIK
```

- Bei dem **SICHERHEITSCODE** handelt es sich um den „normalen“ etracker-Erkennungscode.
- Die **ZIEL_URL** ist eine Redirect-URL, die für das Abbruch-Tracking nicht benötigt wird.
- **BEZEICHNUNG_IN_DER_STATISTIK** ist die Bezeichnung für den Fehler. Dieser sollte aus Gründen der Übersichtlichkeit einer gewissen Notation folgen. Beispiel: **<kennung>--<feldname>-<fehlertyp>** (lasterror--firstname-mandatory).

Um die Daten in unserem Beispiel an den etracker zu senden, kann die Methode sessionDestroyed beispielsweise folgendermaßen überschrieben werden:

```
    @Override
public void sessionDestroyed(HttpSessionEvent
se) {
  String sicherheitsCode = "*****"; // eTra-
cker Code
  String etrackerURL =
http://www.etracker.de/lnkcnt.php?et=
          + sicherheitsCode
          + "&url=http://no-valid-
domain.nc&lnkname=lasterror--";

  // Fehler aus der Session holen
  HttpSession session = se.getSession();
```

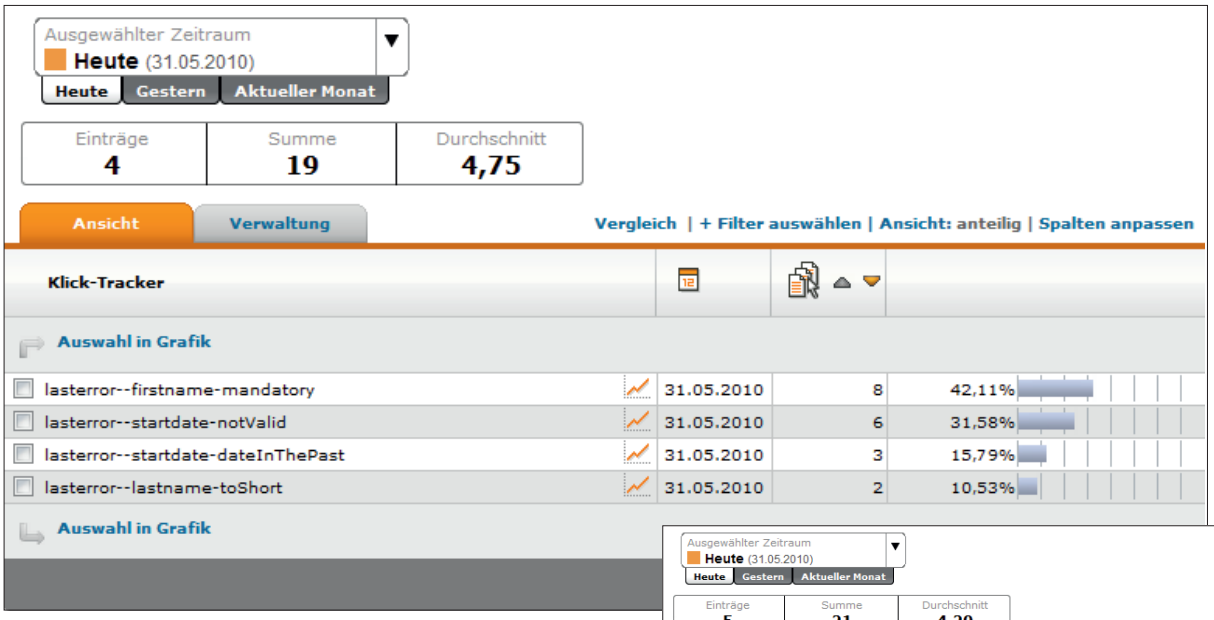

Abbildung 3: Anzeige der Fehlermeldungen im Event-Tracker von etracker

```
  @SuppressWarnings("unchecked")
  List<Pair<String, String>> errors =
                (List<Pair<String, String>>)
session.getAttribute("ERRORS");

  if (errors == null)  return; // no errors -
no tracking

  for (Pair<String, String> error : errors) {
    String errorField = error.getKey();
    String errorValue = error.getValue();

    String errorURL = etrackerURL + errorField
+ "-" + errorValue;
    HttpClient client = new HttpClient();
    GetMethod method = new GetMethod(erro-
rURL);
    try {
      client.executeMethod(method);
    } catch (UnknownHostException e) {
      // domain is not valid.
    } catch (Exception e) {
      // TODO handle exception
      e.printStackTrace();
    }
  }
}
```

Der Aufruf des etracker-Servers greift hierbei auf die httpClient API der Apache Software Foundation zurück (Open Source API, siehe *http://hc.apache.org/httpclient-3.x/*).

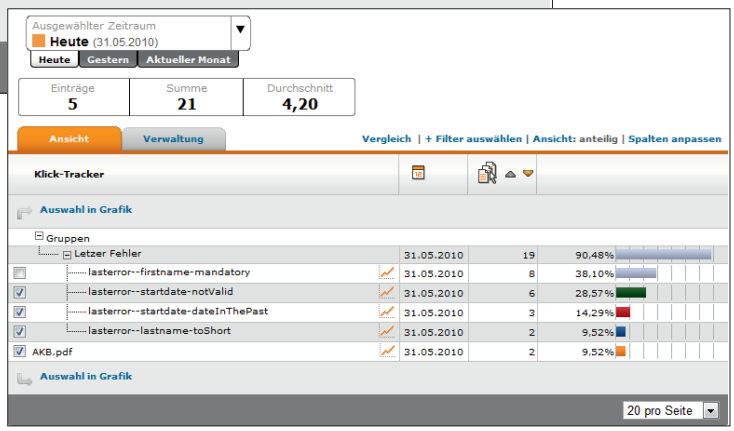

Abbildung 4: Gruppierte Anzeige im Klick-Tracker von etracker

Darstellung im etracker

Im etracker werden diese Meldungen in der Ansicht Nutzung/Klick-Tracker wie in Abbildung 3 zu sehen ist dargestellt.

Wenn der Klick-Tracker auch für seine eigentliche Aufgabe, also das Tracking von Downloads, verwendet wird, kann man hier entsprechende Filter oder Gruppen definieren, die der Übersicht dienen. Ein Beispiel für eine gruppierte Ansicht, die auch Download-Dokumente enthält, zeigt Abbildung 4.

Fazit

Das Tracking von Fehlermeldungen und Abbruchgründen ist eine Voraussetzung für die zielgerichtete Optimierung von Webformularen. Die beschriebenen Analyseverfahren liefern fundierte Informationen über mögliche Fehlerquellen und Stolperfallen bei der Benutzerführung. Auf dieser Basis sollten Unternehmen den Formularprozess dann schrittweise vereinfachen und an die Bedürfnisse der User anpassen. Damit sorgen sie dafür, dass Interessenten, die bereits eine Kaufentscheidung getroffen haben, ihren Bestellvorgang auch unkompliziert beenden können. ¶