HANSER

Leseprobe

Stefan Toth

Vorgehensmuster für Softwarearchitektur

Kombinierbare Praktiken in Zeiten von Agile und Lean

ISBN (Buch): 978-3-446-44395-2

Weitere Informationen oder Bestellungen unter

http://www.hanser-fachbuch.de/978-3-446-44395-2

sowie im Buchhandel.

Inhalt

Geleitwort

Das Märchen vom agilen Architekten

„Heißt du etwa Rumpelstilzchen?“ –
„Das hat dir der Teufel gesagt, das hat dir der Teufel gesagt!“

(Kinder- und Hausmärchen der Brüder Grimm, 7. Auflage 1857)

Die schöne Müllerstochter, die aus Stroh Gold spinnen sollte, hat den Namen von Rumpelstilzchen nicht etwa geraten. Dazu hätte sie mehr als die drei bei den Gebrüdern Grimm beschriebenen Iterationen gebraucht. Sie hatte Wissen (nicht vom Teufel). Und als Ali Baba „Sesam, öffne Dich“ sprach, um in die Höhle mit unermesslichen Schätzen zu gelangen, hat er sich das auch nicht selbst ausgedacht. Er hat es sich abgeguckt von 40 Leuten, die schon mal drin waren in der Höhle. Er konnte auf deren Erfahrung zurückgreifen.

Der Schatz, um den es in diesem Buch von Stefan Toth geht, lässt sich verkürzt als Antwort auf folgende Frage beschreiben: Wie passt Softwarearchitekturmethodik zu einem zeitgemäßen Vorgehen? Oder besser noch: Wie können sie gemeinsam größeren Nutzen bringen?

Dass diese Frage viele bewegt, erlebe ich selbst regelmäßig in Workshops zu meinem Lieblingsthema Architekturdokumentation. Dort geht es darum, wie man Softwarearchitektur nachvollziehbar festhält und kommuniziert; in den Veranstaltungen drehen sich Fragen und Diskussionen regelmäßig darum, ob und wenn ja wie die gezeigten Zutaten zu einem agilen Vorgehen wie beispielsweise Scrum passen. Ganz allgemein können Sie das Interesse aber auch an den zahlreichen Blog- und Konferenzbeiträgen der letzten Jahre ablesen. Diese verknüpfen die Begriffe „agil“ (als griffigstes Wort für zeitgemäßes Vorgehen) und „Architektur“ mal mehr mal weniger pfiffig im Titel, etwa: „Jenseits des Elfenbeinturms – der agile Architekt“ oder „Architektur und agiles Vorgehen – ein Widerspruch?“. Und mehr noch ist es abzulesen an den vollen Sälen, wenn solche Vorträge stattfinden. Die Frage weckt Interesse. Gibt es gute Antworten?

Konferenzbeiträge – zumindest die, die ich gesehen habe – folgten in ihrem Ablauf häufig einem Schema: Zunächst werden die Begriffe „Agilität“ und „Architektur“ ausführlich definiert oder zumindest geklärt. Bei Agilität ist es dabei Folklore, das agile Manifest mit seinen berühmten vier Wertpaaren („Individuen und Interaktionen vor Prozessen und Werkzeugen“ etc.) auf eine Folie zu bannen. Dann wird der angebliche Widerspruch herausgearbeitet, der umso dramatischer ausfällt, je schwergewichtiger und klassischer das Verständnis von Softwarearchitektur, der zugrunde liegende Entwicklungsprozess und die damit verbundenen Artefakte in Notationen der 1990er-Jahre geschildert werden. Schließlich wird der Widerspruch durch sogenannte Best Practices aufgelöst („funktioniert doch super zusammen“).

Wolkige Tipps wie zum Beispiel: kein „Big Upfront Design“, auf die SOLID-Prinzipien achten, die Architektur iterativ und inkrementell entwickeln wie „den Rest“ auch ...

Die Zuhörer verlassen den Saal etwas enttäuscht. Alles richtig, gut und schön, aber wie genau machen wir das jetzt in unserem Projekt? Wo fangen wir an? Wenn schon kein Big Upfront Design, wie klein ist dann das richtige Small? Es liegt wohl auch, aber nicht nur am Format des Frontalvortrags und der oft kurzen Vortragsdauer (beliebt: 45 Minuten), dass die wirklich spannenden Fragen auf Konferenzen oft unbeantwortet bleiben. Mitunter fehlt es auch schlicht an ausreichenden praktischen Projekterfahrungen. Märchenstunde?

Für mich steht außer Zweifel, dass Stefan Toth die nötige Erfahrung besitzt. Er hat sehr unterschiedliche Projekte über einen längeren Zeitraum begleitet und zahlreiche einschlägige Workshops durchgeführt. Bei den Kunden wurde mal klassisch, mal agil, mal irgendwie dazwischen vorgegangen und auch die Branchen könnten unterschiedlicher kaum sein. Vom Finanzsektor bis zur Gaming-Plattform war alles dabei. Das Themenspektrum umfasste die methodische Softwarearchitektur vom Entwurf bis zur Bewertung von konkreten Architekturentscheidungen. So hat Stefan beispielsweise ein agiles Team begleitet und befähigt, regelmäßige Architekturbewertungen in ihren Entwicklungsprozess zu integrieren und eigenverantwortlich durchzuführen. Während viele bei Architekturbewertung sofort an schwergewichtige Methoden wie ATAM denken, wirkt hier nun ein schlankes, aber wirkungsvolles Set an Elementen, bei großer Akzeptanz im Team.

Das ist vielleicht auch schon die Grundidee des Buchs: Es gibt nicht den einen Weg für alle Projekte. Aber es gibt bewährte und schlanke Praktiken in Form von Puzzleteilen, die Nutzen stiften.

In einigen Projekten und Workshop-Situationen, eigentlich in zu wenigen, hatte ich als Kollege das Vergnügen, mit Stefan Toth zusammenzuarbeiten, und konnte wie die Mitarbeiter der Kunden an seinem Wissen und seinen Erfahrungen teilhaben. Und so freut es mich, dass Sie nun als Leser dieses Buchs ebenfalls davon profitieren können.

Denn Stefan Toth hat ein passendes Format zur Vermittlung seines Wissens und Könnens gewählt. Anders als es in einem knappen Vortrag möglich wäre, stellt er hier im Buch seine Ideen ausführlich dar und illustriert sie mit Beispielen. Gleichzeitig ist das Buch lebendig und kein langweiliger Schmöker. Stefan hat viel von seinem Witz in die Zitate und Antipatterns einfließen lassen, ohne dabei albern oder unsachlich zu werden. Die Idee, die einzelnen Zutaten als kombinierbare Muster darzustellen, macht die Inhalte nicht nur leichter erlernbar, sondern vor allem auch einzeln anwendbar. Das erleichtert den Start in Ihrem Projekt ungemein. Die einzelnen Zutaten sind trotzdem kein loses Schüttgut, sondern gut aufeinander abgestimmt und in ihrer Gesamtheit schlüssig. Ausdrucksstarke Visualisierungen – eine besondere Spezialität von Stefan – vermitteln komplizierte Inhalte gut erinnerbar und verknüpfen die einzelnen Muster.

Aus eigener Erfahrung kann ich sagen, dass die Erarbeitung und Aufbereitung von Inhalten in Form eines Buchs große Vorteile bietet (die hier auch ausgeschöpft wurden), aber auch einen nicht zu unterschätzenden Nachteil, zumindest verglichen mit Vorträgen oder einem Workshop. Es besteht die Gefahr, dass man als Autor weniger Feedback bekommt. Ich möchte Sie daher ermutigen, Erfahrungen, die Sie mit den dargestellten Praktiken machen konnten, zu teilen. Tauschen Sie sich aus, mit dem Autor und auch mit anderen Lesern.

Um zum Schluss noch mal auf Rumpelstilzchen zurückzukommen: In diesem Buch lernen Sie nicht, wie Sie aus Stroh Gold spinnen. Dafür viele andere Dinge, die Sie jetzt vermutlich

auch noch nicht können. Und es ist kein Märchen. Alles ist wahr. Wenn Sie mögen, schließen Sie das Buch nun kurz, sprechen mir nach: „Sesam, öffne Dich“, und schlagen es wieder auf. Und es tut sich tatsächlich ein reicher Schatz an Erfahrungswissen auf, der nur darauf wartet, Stück für Stück heraus in Ihr Projekt getragen zu werden. Mir bleibt nur noch, Ihnen viel Freude damit zu wünschen.

Stefan Zörner

5.1 Informativer Arbeitsplatz

„Die Wahrheit ist immer konkret."

– Lenin

Peter: Was habt ihr gestern eigentlich besprochen in dem Treffen?

Michael: Ach, Tommy wollte Verantwortlichkeiten in das Archivsystem verschieben, wegen befürchteten Wartungsproblemen und zyklischen Abhängigkeiten.

Peter: Ich fände gut, wenn ihr so etwas irgendwo dokumentiert. Ich habe auch mit dem Archivsystem zu tun und würde ungern Dinge bauen, die es vielleicht schon gibt.

Michael: Verstehe ich schon. Es war aber nur ein kurzes Meeting. Wir haben eine Notiz ins Wiki gelegt.

Peter: Im Wiki sind so viele alte und falsche Informationen, da suche ich nicht unbedingt nach neuen Ideen. Wir sollten Arbeit an der Architektur allgemein sichtbarer machen. Auch die Prinzipien, die wir letztens beschlossen haben, müssten doch alle kennen.

Michael: Welche Prinzipien?

Peter: Siehst du, das meine ich! Wir könnten die große Projektwand von den Motivationspostern befreien und neben unsere Architekturziele auch solche Informationen hängen. Fände ich gut.

Michael: Ja, warum nicht. Die Motivationsposter sind sowieso ein schlechtes Zeichen für die meisten Projekte. „Wer das Beste versucht, dem gelingt alles" ... mhm ...

Architekturentscheidungen sind weitreichend und vor allem deshalb schwer änderbar, weil weite Systemteile davon betroffen sind. Durch die breite Wirkung sind auch viele Entwickler an der Umsetzung einer Architekturentscheidung beteiligt, sie müssen sich entsprechend der Architekturidee verhalten und untaugliche Entscheidungen früh identifizieren können. Sie wollen Architekturideen deshalb möglichst breit streuen und in vielen Köpfen verankern. Entwickler brauchen für gute Arbeit Architekturkontext und Transparenz.

Problemstellung:

Wie können wichtige Informationen zur Architektur möglichst breit gestreut werden, um (1) Kontext für Entwurf und Entwicklung zu geben und (2) bei schwierigen Entscheidungen und Kompromissen für eine gemeinsame Basis zu sorgen?

Jeder Entwickler, Berater oder andere an der Umsetzung Beteiligte sollte sich schnell einen Überblick zur Softwarearchitektur verschaffen können, sollte wissen, wie die Architektur grob aussieht, was sie formt und wie es um sie steht. Zentral sind folglich das sogenannte Big-Picture, wichtige Entscheidungen, die stärksten Einflüsse und der aktuelle Status.

Das Konzept, Information sichtbar zu machen, sie an Plätzen zu „deponieren“, die viel frequentiert sind, und sie so als allgemeines Wissen in den Köpfen zu verankern, ist nicht neu. Verbunden mit „beobachtenden Augen“ ist es das Mittel von Propaganda in fragwürdigen Diktaturen dieser Erde (und es funktioniert nachweisbar seit Jahrzehnten). Das ist aber nicht das Bild, das ich hier vermitteln möchte. Vielmehr möchte ich auf die Disziplin des Projektmanagements verweisen, die ebenfalls schon lange mit diesem Mittel arbeitet. 2004 hat Kent Beck die Idee für eXtreme Programming formuliert und dabei vom *informativen Arbeitsplatz* gesprochen. Betritt ein interessierter Beobachter den Projektraum, soll er demnach innerhalb von 15 Sekunden eine generelle Idee haben, wie es um das Projekt steht. Nimmt er sich etwas mehr Zeit für einen genaueren Blick, offenbaren sich ihm vorhandene oder potenzielle Probleme [Bec04]. Seitdem arbeiten agile Projekte mit transparenten Prozessen, Backlogs an Projektwänden, Kanban-Boards, plakatierten Release-Zielen, Burndown-Charts, sichtbarem Build-Status und vielem mehr. Elizabeth Whitworth hat in einer Studie belegt, dass ein informativer Arbeitsplatz positive Auswirkungen auf Motivation, Stolz und Begeisterung der Mitarbeiter hat [Whi06] – neben der besseren Handlungsfähigkeit durchaus erstrebenswerte Effekte. Was liegt also näher, als diese Erfolge auf Softwarearchitektur zu übertragen – einer Disziplin, die sich über ihre querschnittliche, fundamentale Wirkung auf viele Beteiligte definiert?

Was können Sie nun tun, um einen informativen Arbeitsplatz für Architektur aufzubauen? Die grundsätzlichen Elemente sind die folgenden:

- **Informationsverteiler** (engl. information radiators): manuell erstellte und aktuell gehaltene, nichtelektronische Anzeigen und Darstellungen, grob gesagt: Wände und Flipcharts.
- **Feedback-Geräte** (engl. feedback devices): Geräte, die mit automatisierten Prozessen verbunden sind und deren Status zeigen. Üblich sind etwa Bildschirme oder Ampeln.

Informationsverteiler sind nicht an automatisierte Prozesse gebunden und sehr einfach und universell zu verwenden. Hier ist wenig Vorbereitung nötig, Sie können sofort loslegen. Plakatieren Sie wichtige Architekturinformationen an die Wände des Büros und sorgen Sie dafür, dass Architekturtreffen sichtbare Spuren hinterlassen (siehe Ad-hoc-Architekturtreffen → Abschnitt 4.8). Die Darstellung sollte deutlich und auch aus einigen Metern Entfernung gut lesbar sein. Was sind nun aber jene Architekturinformationen, die wichtig sind? Bild 5.2 gibt einen Überblick.

Die illustrierten Elemente sind sinnvolle Beispiele für architektonische Inhalte von Informationsverteilern:

- **Was die Architektur formt**:
  Stellen Sie die wichtigsten Einflüsse auf die Architekturarbeit möglichst übersichtlich dar. Geeignet sind hierfür etwa der *Systemkontext*, der Fremdsysteme und Benutzer Ihres Systems zeigt, der *Produktkarton*, der Ihre Vision und die wichtigsten Qualitätseigenschaften vermittelt, die Top-drei- bis -fünf-*Architekturziele* und die momentan wichtigsten *Szenarien* – jene, die technisch herausfordernd, für den Kunden wichtig und momentan dringend sind.
- **Wie die Architektur grob aussieht**:
  Zeigen Sie ein übersichtliches *Big-Picture* Ihrer Anwendung, das zumindest die zentralen Muster, Technologien und Komponenten beinhaltet. Machen Sie die wichtigsten getroffenen *Architekturentscheidungen* deutlich (siehe Architekturentscheidungen treffen → Abschnitt 4.4) und präsentieren Sie die entschiedenen *Architekturprinzipien* zumindest mit aussagekräftigen Namen (siehe Im Prinzip entscheiden → Abschnitt 4.7).

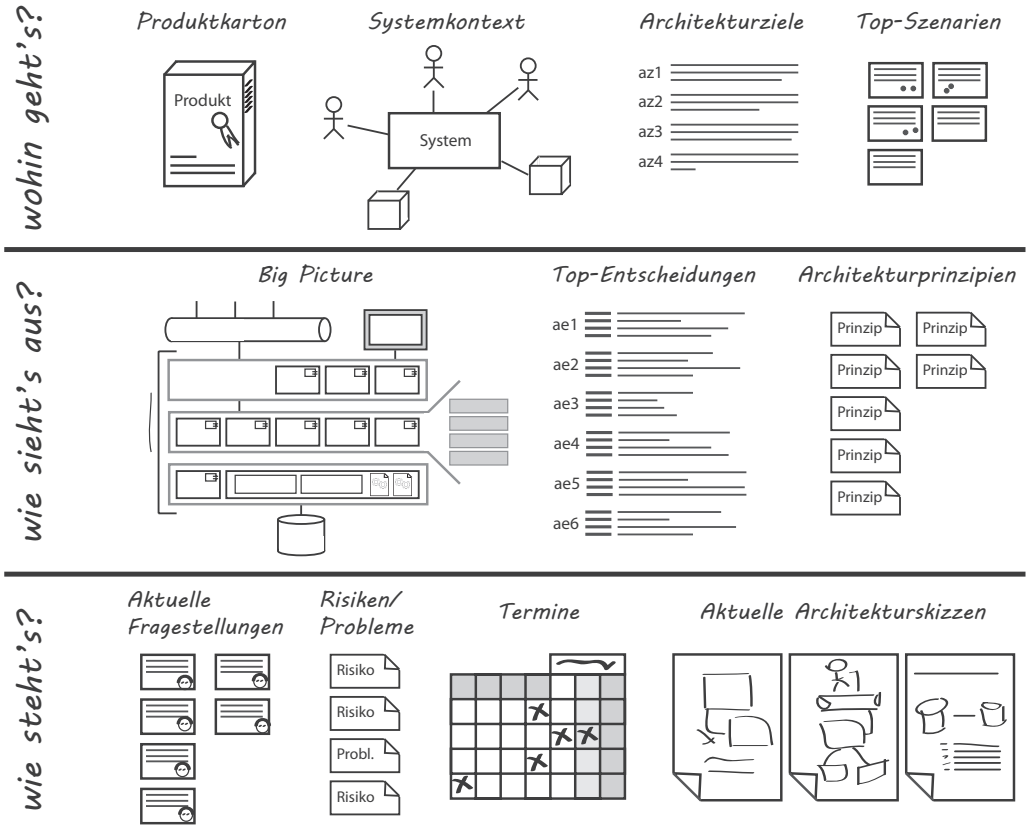

BILD 5.2 Inhalte einer „Architekturwand“

- **Wie es um die Architektur steht**:
  Beim aktuellen Status zur Architektur sind zum Beispiel Flipcharts mit *Architekturskizzen* (etwa aus AD-HOC-ARCHITEKTURTREFFEN → Abschnitt 4.8) und aktuell bearbeitete *Fragestellungen* interessant, außerdem erkannte und noch nicht behandelte *Risiken* oder *Probleme* (aus der wiederholten Reflexion, dem Realitätscheck für Architekturziele oder einfach aus der Umsetzungsarbeit). Auch wichtige *Termine* von Abstimmungen, Treffen und Demos sind spannend.

Das Wichtigste bei Informationsverteilern ist, sie aktuell und lebendig zu halten. Übertragen Sie die Aufgabe nicht einer Person, sondern machen Sie alle Entwickler für die „Architekturwand“ verantwortlich. Aktualisieren Sie die Architekturziele nach REALITÄTSCHECKS (→ Abschnitt 6.2), aktualisieren Sie die wichtigsten Szenarien bei der ANFORDERUNGSPFLEGE (→ Abschnitt 3.2), achten Sie auf Entscheidungen bei der WIEDERKEHRENDEN REFLEXION (→ Abschnitt 5.5). Veränderungen am Big-Picture sollten Sie einzeichnen und z. B. mit roten Post-Its signalisieren. Lassen Sie wertvolle Flipcharts nicht einfach in Besprechungsräumen hängen, sondern stellen Sie sie einige Tage an zentraler Stelle aus. Probleme und Risiken lassen sich auf Post-Its sammeln und so weiter. Eine gute Architekturwand kann vielen Missverständnissen vorbeugen und motiviert Entwickler dazu, sich mit architektonischen Ideen auseinanderzusetzen.

Verteilt sich Ihre Entwicklungsmannschaft auf mehrere Räume, können Sie versuchen, einige Informationen zu doppeln (vorrangig Einflüsse und Architekturüberblick) und andere spezifisch für die Mitarbeiter vor Ort zu gestalten (etwa den Status). Redundanzen haben allerdings immer ihre Schattenseiten. Bevor Sie gänzlich auf eine physische Wand verzichten und mit elektronischen Dashboards, Wikis und Bildschirmschonern arbeiten: Ein Projekt, das ich betreut habe, hat erfolgreich mit einer physischen Architekturwand gearbeitet und diese Wand in drei andere Projekträume „übertragen" – mit fest installierter Kamera jeweils ein bis zwei Beamern in den weiteren Räumen. Das Experiment hat überraschend gut funktioniert.

Feedback-Geräte werden üblicherweise eingesetzt, um den Status von Build und Tests anzuzeigen. Neben der berühmten Build-Ampel können z. B. Bildschirme zum Einsatz kommen, die in großer Schriftgröße und farblich unterlegt Testergebnisse und Erfolgsraten kommunizieren. Aus der Perspektive der Softwarearchitektur betrachtet, sind neben diesen beiden Klassikern auch Ergebnisse aus der statischen Codeanalyse (Qualitätsindikatoren, siehe Qualitätsindikatoren nutzen → Abschnitt 6.4), dem Vergleich zwischen Code und Architektur (→ Abschnitt 6.5) und aus späteren Testphasen für qualitative Aspekte (siehe qualitative Eigenschaften testen → Abschnitt 6.3) interessant. Etwas ausgefallener ist die Idee auf einem Bildschirm Fotos von den letzten Architektur-Workshops zu zeigen. Laden Sie etwa Fotos von Whiteboard-Skizzen ins Wiki, werden Sie automatisch an Bildschirmen in den Projekträumen gezeigt, inklusive Workshop-Titel und Namen der Teilnehmer.

Tipps zum Scheitern

Möchten Sie mit diesem Muster scheitern und Ihrem Projekt schaden, folgen Sie diesen Empfehlungen:

- Lassen Sie Informationen auf Informationsverteilern ruhig veralten. Wenn sich an der Architekturwand lange nichts ändert, achtet auch keiner darauf – das reguliert sich von selbst.
- Wählen Sie für Informationsverteiler eine Wand in einem entfernten Besprechungsraum. Interessierte können dort nachsehen und die Architekturwand ist vor neugierigen Blicken des Managements und fremder Projekte geschützt.
- Ernennen Sie einen Entwickler zum Informationsminister. Er ist als Einziger befugt, etwas an Informationsverteilern zu ändern oder zu aktualisieren.
- Verzichten Sie darauf, Namen auf Flipcharts und Post-Its mit Risiken und Problemen zu schreiben. Die Erarbeiter sind so vor lästigen Fragen geschützt und können in Ruhe arbeiten.
- Gestalten Sie nicht eine Architekturwand, sondern viele. Alle Skizzen und Diagramme bleiben an den Wänden, bis der Platz gebraucht wird. So sind überall wertvolle Informationen und der informative Arbeitsplatz wird größer!
- Verteilen Sie alle Test- und Metrikergebnisse feingranular und mehrmals täglich als E-Mail an alle Mitarbeiter. Im Englischen sagt man „hard to avoid" – das rückt die Wichtigkeit des Themas ins Bewusstsein aller ... oder so ähnlich.

Der informative Arbeitsplatz trägt die wichtigsten Architekturinformationen zusammen und ist deshalb sowohl Informationsquelle als auch Ergebnissenke. Die Inhalte für Informationsverteiler kommen zum Beispiel aus dem INITIALEN ANFORDERUNGS-WORKSHOP (→ Abschnitt 3.1 – Architekturziele, Produktkarton etc.), aus der ANFORDERUNGSPFLEGE (→ Abschnitt 3.2 – Szenarien), aus AD-HOC-ARCHITEKTURTREFFEN (→ Abschnitt 4.8 – Architekturskizzen) oder dem REALITÄTSCHECK FÜR ARCHITEKTURZIELE (→ Abschnitt 6.2 – Probleme und Risiken). Außerdem erzeugen Sie Inhalte, wenn Sie GERADE GENUG ARCHITEKTUR VORWEG entscheiden (→ Abschnitt 4.3 – Systemkontext, Architekturziele, Szenarien), ARCHITEKTURENTSCHEIDUNGEN TREFFEN (→ Abschnitt 4.4 – Entscheidungen, Risiken) oder IM PRINZIP ENTSCHEIDEN (→ Abschnitt 4.7 – Prinzipien).

Die meisten dieser Muster bedienen sich auch wiederum an „der Wand“. So holt sich der REALITÄTSCHECK FÜR ARCHITEKTURZIELE (→ Abschnitt 6.2) etwa die Ziele von der Architekturwand. Darüber hinaus können Sie entdeckte Probleme oder RISIKEN AKTIV BEHANDELN (→ Abschnitt 4.6).

Feedbackgeräte basieren darauf, dass sie KONTINUIERLICH INTEGRIEREN UND AUSLIEFERN (→ Abschnitt 6.6), QUALITATIVE EIGENSCHAFTEN TESTEN (→ Abschnitt 6.3), QUALITÄTSINDIKATOREN NUTZEN (→ Abschnitt 6.4) und CODE UND ARCHITEKTUR VERBINDEN (→ Abschnitt 6.5). Die entsprechenden Reports aus Test- und Metrikergebnissen werden entsprechend aufbereitet, bevor sie zur Anzeige gebracht werden.

Stichwortverzeichnis