Software Engineering

Grundlagen, Menschen, Prozesse, Techniken

DAS VORWORT

Vorwort

Entstehungsgeschichte

Dieses Buch hat seine allerersten Anfänge in einer Vorlesungsreihe »Software Engineering« am Neu-Technikum in Buchs, St. Gallen, Schweiz (Ludewig, 1985). Das Material wurde durch viele weitere Vorlesungen beider Autoren an der ETH Zürich, der Universität Stuttgart und der RWTH Aachen University immer wieder bearbeitet, ergänzt und erweitert.

2006 endlich ging unser Buch in Druck. 2010 erschien die zweite, 2013 die dritte Auflage. Nun, 2022, war es an der Zeit, die vierte Auflage fertigzustellen. Nachdem in den Auflagen 2 und 3 Korrekturen und kleine Ergänzungen im Vordergrund standen, gibt es nun einige substanzielle Erweiterungen, vor allem bei den Software-Prozessen und im Software-Entwurf.

Die lange Geschichte des Buches steht scheinbar im Widerspruch zum raschen Verderb des Informatik-Wissens. Wir sehen das weniger ängstlich und halten das Gerede von der »kurzen Halbwertzeit des Wissens« für modisches Geschwätz. Da wir uns nicht mit dem Leistungsstand von Mikroprozessoren befassen, sondern die Leserinnen und Leser, soweit es in unseren Kräften steht, mit dem Grundwissen für ein hoffentlich langes und befriedigendes Berufsleben ausrüsten, sollte das hier präsentierte Material viele Jahre brauchbar bleiben. Dazu trägt ohne Zweifel bei, dass sich Software Engineering vor allem mit den Menschen befasst, die Software in Auftrag geben, entwickeln und ändern oder benutzen. Was die Philosophen vor mehr als zweitausend Jahren über Menschen gesagt haben, gilt zum größten Teil noch heute, und so wird es wohl noch ein paar weitere Jahre gültig bleiben.

Rollenbezeichnungen

Auch in diesem Buch bleibt das Problem ungelöst, eine befriedigende Form der Rollenbezeichnungen zu finden, die nicht suggeriert, dass die Person in dieser Rolle ein männliches, ein weibliches oder ein diverses Wesen ist. Wir haben uns an anderer Stelle mit diesem Problem näher befasst (Deininger et al., 2017), freilich ohne eine gute, allgemein akzeptierte Lösung zu finden.

Darum gehen wir den üblichen Weg, alle Rollenbezeichnungen in ihrer männlichen Grundform, dem generischen Maskulinum, zu verwenden. Es dürfte überflüssig sein, darauf hinzuweisen, dass es keine einzige Rolle auf dem Gebiet des Software Engineerings gibt, die vorzugsweise oder ausschließlich mit Personen eines bestimmten Geschlechts besetzt sein sollte. Nach unserer Erfahrung sind gemischte Teams immer den ungemischten vorzuziehen.

Sprachstil

Vor drei, vier Jahrzehnten wäre eine Vorbemerkung zum Sprachstil nur töricht gewesen. Bis auf wenige bekannte Riffe im »Wörtersee« war allgemein klar und anerkannt, was als Deutsch gilt.

Das ist heute nicht mehr so. Ursache ist anscheinend nicht die Rechtschreibreform, sondern ein dramatischer Verfall der sprachlichen Disziplin. Viele Texte, die von Studierenden abgeliefert werden, können kaum mehr als Deutsch bezeichnet werden, und viele derer, die eigentlich Vorbild sein sollten, also Journalisten, Politiker und natürlich auch Hochschullehrer, verstärken das Problem, statt es zu bekämpfen.

Wir bekennen uns ausdrücklich zum Ziel, unsere Gedanken in einer akzeptablen Sprache zu formulieren. Wir sind dabei vor Fehlern nicht sicher, und wir werden keine literarische Qualität erreichen, aber wir bemühen uns. Den Leserinnen und Lesern wird darum auch an einigen Stellen ein Sprachgebrauch auffallen, der unmodern erscheint, aber – wenigstens in unseren Ohren oder Augen – richtig ist.

Anglizismen lassen sich gerade in Informatik-Büchern nicht vermeiden; das beginnt schon mit unserem Fachgebiet und mit dem Titel dieses Buches. Wo es eine gute deutsche Übersetzung gibt, ziehen wir sie vor. Gerade beim »Software Engineering« ist das zweifelhaft, denn »Softwaretechnik« ist ein sprachlicher Zwitter. Leider wurde versäumt, das Wort »Software« in unsere Sprache zu übertragen, wie es den Franzosen mit »Logiciel« glänzend gelungen ist. Dass solche Wortschöpfungen auch im Deutschen möglich sind, zeigt sich an künstlichen, aber heute geläufigen Wörtern wie »Rechner« oder »Datei«.

Dank

»Wir sind Zwerge, die auf den Schultern von Riesen sitzen.« So hat es laut Wikipedia Bernhard von Chartres um 1120 formuliert. Das gilt natürlich vor allem für die Verfasser wissenschaftlicher Arbeiten. Wo wir bewusst auf Material Bezug nehmen, das von anderen Autoren stammt, haben wir (hoffentlich vollständig und korrekt) zitiert. Das ist in besonders hohem Maße bei zwei Büchern der Fall, an denen einer von uns beteiligt ist (Frühauf, Ludewig, Sandmayr, 2002 und 2007). Vor allem unsere Kapitel 13 (*Software-Qualitätssicherung und -Prüfung*), 18 (*Programmtest*), 20 (*Konfigurationsverwaltung*) und 21 (*Software-Wartung*)

enthalten Anleihen aus diesen Büchern. Herzlichen Dank an die beiden Kollegen in Baden/Schweiz, die mit ihrer INFOGEM AG gutes Software Engineering erfolgreich vermitteln und praktizieren!

Einige Koryphäen des Software Engineerings haben uns fast ein halbes Jahrhundert lang geprägt: Das waren vor allem David L. Parnas (Jahrgang 1941), Barry W. Boehm (1935–2022), Fred Brooks (1931–2022) und Michael Jackson (1936), auch heute kaum noch bekannte Pioniere wie Daniel Teichroew (1925–2003), der das erste Spezifikationssystem geschaffen hat (PSL/PSA). In Deutschland ist vor allem Friedrich L. Bauer (1924–2015) zu nennen, der den Begriff »Software Engineering« 1968 wenn nicht erfunden so doch populär gemacht hat. Ohne diese (und viele andere) Leute wäre das Fachgebiet deutlich ärmer und ein Lehrbuch darüber ein »Leerbuch«.

Wenn es Unklarheiten über Begriffe oder Quellen gibt, schaut man ins Internet – und findet dort sehr viel Müll. Umso erfreulicher sind die Ausnahmen; Martin Glinz (bis zur Emeritierung 2017 an der Universität Zürich) sei stellvertretend für alle genannt, die Qualitätssicherung nicht nur lehren, sondern auch leben und (z.B. durch ihre Webseiten) demonstrieren.

Kornelia Kuhle hat (bis zur dritten Auflage) mit scharfem Blick auch kleinste Unregelmäßigkeiten des Textes erkannt und markiert. Einzelne Kapitel wurden von unseren Mitarbeiterinnen und Mitarbeitern kommentiert. Ihnen allen herzlichen Dank! Und allen, die uns zugehört und auch widersprochen haben, sind wir zu großem Dank verpflichtet.

Die Zusammenarbeit mit dem dpunkt.verlag, stets bestens vertreten durch Christa Preisendanz, war über die vielen Jahre reibungslos und immer erfreulich. Ursula Zimpfer hat wieder einmal mit bewunderungswürdiger Präzision die Endkontrolle des Buches vorgenommen; was jetzt noch falsch ist, geht auf unsere Kappe.

Unseren Familien danken wir dafür, dass sie uns immer den nötigen Freiraum und die Zeit gewährt haben, um an diesem Buch zu arbeiten.

Natürlich freuen wir uns auch in Zukunft über Ihre Kritik, positiv oder negativ, und Ihre Hinweise auf Lücken und Mängel. Dafür schon heute vielen Dank!

Jochen Ludewig, Horst Lichter
Stuttgart und Aachen, Dezember 2022

Inhalt und Aufbau, Zielgruppen

Inhalt

Vier Kriterien entscheiden darüber, welche Themen in einem Lehrbuch behandelt werden:

- Welches Wissen bringen die Autoren mit?
- Welche Themen sind für die Leser wichtig und attraktiv?
- Welche Aussagen und Erkenntnisse sind mutmaßlich für einige Jahre stabil, nicht den kurzfristigen Moden unterworfen?
- Was kann und sollte in einer Vorlesung über Software Engineering behandelt werden?

Jedes der Kriterien definiert, mehr oder minder scharf, eine Menge; die Schnittmenge liefert den Themenkatalog, der dem Buch zugrunde liegt.

Dabei müssen auch – wie immer im Software Engineering – Kompromisse gefunden, also Einschränkungen beim einen oder anderen Kriterium in Kauf genommen werden. Für die Leserinnen und Leser sollte aber in allen Teilen deutlich werden, welches Verständnis wir von unserem Fach haben: Software Engineering ist eine auf der Informatik beruhende Ingenieurdisziplin, die wie alle Ingenieurfächer darauf abzielt, die Praxis zu verbessern.

Software Engineering stellt sich für uns wie ein weitgehend unerforschter Kontinent dar. Wir sind weit davon entfernt, eine vollständige und präzise Beschreibung anbieten zu können. Wenigstens aber die Konturen, die bisherige Entwicklung (in groben Zügen) und die als gesichert geltenden Einsichten soll dieses Buch abdecken. Wer es in der Lehre einsetzt, wird eigene Erfahrungen und spezielle Literatur hinzufügen.

Aufbau

Für den Aufbau eines Lehrbuches gibt es einige sinnvolle Prinzipien:

- vom Allgemeinen zum Speziellen
- vom Leichten zum Schwierigen
- von den Grundlagen zu den Anwendungen

Im Software Engineering kommt noch hinzu:

- dem Gang des Software-Projekts folgend

Leider lässt sich daraus keine konkrete Reihenfolge ableiten, denn diese Prinzipien haben unterschiedliche Konsequenzen. Zudem sind viele Themen zyklisch verbunden. Ein typisches Beispiel ist die Software-Wartung: Im Projektablauf steht sie ganz am Ende. Da aber die Schwierigkeiten der Wartung sehr stark von den Entscheidungen bei der Konstruktion der Software beeinflusst sind, sollten Überlegungen zur Wartung nicht erst angestellt werden, wenn die Entwicklung abgeschlossen ist, sondern bereits in der Projektplanung. Die Wartung wirkt sich also auf den Entwurf aus, weil sich der Entwurf auf die Wartung auswirkt.

Wir können dieses Dilemma nicht auflösen. Wir haben darum einen anderen Aufbau gewählt: In sechs Teilen wird das Thema aus verschiedenen Perspektiven betrachtet.

- Teil I: *Grundlagen*
  Die Inhalte dieses Teils sind fundamental und damit nicht von einer speziellen Technologie geprägt. Hier geht es um das Basiswissen des Software-Ingenieurs. Am Anfang steht ein Kapitel über Modelle, weil dieses Thema für das Software Engineering wirklich grundlegend ist und damit das Fundament aller weiteren Kapitel darstellt.
- Teil II: *Menschen und Prozesse*
  Software ist enger als andere technische Artefakte mit dem Denken der Menschen verknüpft, die die Software herstellen oder benutzen. Die organisatorischen Rahmenbedingungen haben großen Einfluss auf den Erfolg der Projekte. Diese Punkte werden im Teil II behandelt.
- Teil III: *Daueraufgaben im Software-Projekt*
  Viele Tätigkeiten wie Dokumentation oder Prüfung können nicht einzelnen Schritten der Entwicklung oder speziellen Dokumenten zugeordnet werden, sie finden laufend statt. Teil III behandelt diese Daueraufgaben.
- Teil IV: *Techniken der Software-Bearbeitung*
  Die einzelnen Schritte der Entwicklung, von der Analyse bis zur Integration, werden im Teil IV erläutert.
- Teil V: *Verwaltung und Erhaltung von Software*
  Die Wartung ist eng verknüpft mit der Konfigurationsverwaltung, dem Reengineering und der Wiederverwendung. Darum werden alle diese Themen im Teil V des Buches behandelt. Untereinander ist die Abgrenzung unklar; beispielsweise wird die Änderungsverwaltung im Kapitel über die Wartung besprochen, sie könnte ebenso gut im Kontext der Konfigurationsverwaltung behandelt werden.

Die Themen dieses Teils könnten auch dem Teil III, den Daueraufgaben, zugeordnet werden. Hier handelt es sich aber um Aufgaben und Arbeiten, die ganz oder vorwiegend anfallen, nachdem die Software an den Kunden ausgeliefert ist.

- Teil VI: *Software Engineering lehren*
  Am Ende des Buches befassen wir uns mit der Frage, wie sein Inhalt in Studiengängen und in anderen Formen vermittelt werden kann. Dazu wird ein konkreter Studiengang kurz vorgestellt, und es werden einige Zahlen zur Verbreitung solcher Studiengänge angegeben. Auf Kurse und Seminare zu Teilgebieten des Software Engineerings wird kurz hingewiesen.
- Teil VII: *Literatur und Index*
  Die im Buch zitierte Literatur haben wir nach Verfassern, die zitierten Normen nach Normenreihen geordnet; wir haben uns sehr nachdrücklich bemüht, alle Quellen richtig und vollständig anzugeben. Das Stichwortverzeichnis steht ganz am Schluss des Buches.

Wer dieses Buch im Unterricht einsetzt, sollte das Inhaltsverzeichnis nicht als Vorgabe für die Gliederung seiner Lehrveranstaltung betrachten. Wir hoffen, unsere Kolleginnen und Kollegen durch eine klare und nachvollziehbare Struktur bei der individuellen Auswahl der Themen und bei der Gestaltung ihrer Lehre zu unterstützen. Und natürlich gibt es Bedarf und Raum für Ergänzungen durch andere Themen, die in diesem Buch fehlen oder nur kurz besprochen werden. Die formale Spezifikation und eine ganze Reihe moderner Entwicklungstechnologien sind naheliegende Beispiele solcher Gebiete.

Zielgruppen

Da wir regelmäßig Lehrveranstaltungen über Software Engineering an zwei deutschen Universitäten durchführen bzw. durchgeführt haben, sind Studierende die Adressaten dieses Buches. Unsere Erfahrungen in anderen Ländern und in anderen Lehranstalten legen die Vermutung nahe, dass das Buch für alle deutschsprachigen Länder und auch für andere Hochschulen geeignet ist. Dazu zählen wir auch Schulungseinrichtungen in der Industrie, wo wir selbst immer gern unterrichtet haben und unterrichten.

Aus dieser Erfahrung und den Erfahrungen aus vielen Kooperationen mit Industrieunternehmen wissen wir, dass es in der Industrie sehr viele Menschen gibt, die an Software arbeiten, ohne eine einschlägige Ausbildung zu haben. Die meisten von ihnen sind gelernte Ingenieure, Naturwissenschaftler, Mathematiker oder Kaufleute; sie bringen durch ihren ursprünglichen Beruf wichtige Voraussetzungen für das Software Engineering mit und haben Studierenden sehr viel praktische Erfahrung voraus. Dieses Buch gibt ihnen die Möglichkeit, einige Grundlagen und spezielle Themen im Selbststudium zu erarbeiten. Wir haben uns besonders für diese Gruppe darum bemüht, die Ideen und Gedanken nicht nur

aufzulisten, sondern durch einen hoffentlich gut strukturierten und formulierten Text auch verständlich und angenehm lesbar zu machen, also quasi eine Vorlesung in Buchform anzubieten.

Wir haben zu diesem Buch eine Webseite (*se-buch.de*) erstellt. Dort finden Sie unter anderem Materialien, beispielsweise alle im Buch enthaltenen Abbildungen, und auch Korrekturen zu gemeldeten Fehlern.