Physikalische Chemie, Universität des Saarlandes

Vorlesung: Programmierkonzepte in den Naturwissenschaften im Sommersemester 2021

PD Dr. Till Biskup

— Glossar zu Lektion 09: „Bugverwaltung“ —

Hinweis: Die nachfolgend genannten Begriffe und Definitionen erheben keinen Anspruch auf formale Korrektheit, sondern dienen lediglich dem besseren Verständnis der in der Vorlesung behandelten Themen und sind im jeweiligen Kontext zu sehen. Mehrfache, voneinander abweichende Definitionen in unterschiedlichen Kontexten sind daher möglich. Englische Begriffe werden zwar nach Möglichkeit übersetzt, erscheinen aber ggf. unter ihrem englischen Namen in der Liste. Verweise untereinander sind durch ↑ gekennzeichnet.

Berechenbarkeit Formulierbarkeit einer Berechnungsanweisung (Algorithmus) für eine mathematische Funktion

Bug (engl. für „Wanze, Käfer“) Programmfehler oder Softwarefehler, allgemein ein Fehlverhalten von Computerprogrammen.

Bug Tracker ↑Bug-Verwaltung

Bug-Verwaltung engl. *bug tracker*; Software, die spezifisch auf die Handhabung von Fehlern (↑Bug) in Programmen zugeschnitten ist. Strukturiert und vereinfacht den Umgang mit Fehlermeldungen (*bug report*), dokumentiert aufgetretene Fehler und entkoppelt Nutzer und Entwickler zeitlich und räumlich. Darüber kann sie der Diskussion der weiteren Entwicklung dienen, das ist aber eher der Bereich eines ↑Issue Trackers oder dedizierter Projektmanagement-Software.

call stack Aufrufstapel, Stapelspeicher, der zur Laufzeit eines Programms Informationen zu den gerade aufgerufenen Unterprogrammen enthält. Wird oftmals im Falle eines schwerwiegenden Fehlers im Programmablauf ausgegeben und erlaubt nachzuvollziehen, in welcher Reihenfolge Unterprogramme (Funktionen, Routinen) aufgerufen wurden, als der Fehler auftrat.

Coding Conventions Innerhalb der Entwicklergruppe eines Projektes zumindest zu einem gegebenen Zeitpunkt festgelegte Konventionen zu bestimmten Aspekten der Formatierung von Quellcode. Dient der Vereinheitlichung und trägt dadurch wesentlich zur Lesbarkeit bei. Wichtiger als der Inhalt ist die konsequente Befolgung der Konventionen und der möglichst breite Konsens über die Inhalte innerhalb der Entwicklergruppe.

Code-Analysator Werkzeug zur ↑Codeanalyse, i.d.R. zur ↑statischen Codeanalyse.

Codeanalyse i.d.R. zumindest teilweise automatisch durchgeführte Analyse des Quellcodes einer Software vor oder während ihrer Ausführung. Die Analyse ohne Ausführung des Quellcodes heißt ↑statische Codeanalyse, jene während der Ausführung ↑dynamische Codenanalyse. Eng verwandt mit dem systematischen Testen von Software (vgl. ↑Unittest).

Code Review Gemeinsame Begutachtung eines Quellcode-Abschnitts eines Projektes durch mehrere Projektbeteiligte. Ein Code Review ist formaler als das ↑Pair Programming, außerdem wird beim Code Review in der Regel *nicht* programmiert, sondern nur über den Quellcode diskutiert. Ziele eines Code Reviews sind u.a.: Wissensvermittlung innerhalb des Teams, Sicherstellen eines gemeinsamen

Kenntnisstands, Entwicklung von Ideen für die Weiterentwicklung. Eine Verbesserung des diskutierten Codes geht meist damit einher, allerdings sollte Kritik immer konstruktiv sein.

design by contract Vor- und Nachbedingungen für Codeblöcke werden explizit im Quellcode festgehalten und lassen sich so vom Compiler und zur Laufzeit überprüfen. Wird nur von wenigen Programmiersprachen konsequent unterstützt (z.B. ADA/SPARC, Eiffel), die meist in sicherheitskritischen Anwendungen (z.B. Flugsicherung/Luftraumüberwachung, Raumfahrt) eingesetzt werden.

dynamische Codeanalyse ↑Codeanalyse während der Ausführung des Quellcodes. Ein Programm oder Teile davon werden kontrolliert ausgeführt, um so mögliches Fehlverhalten aufzuspüren. Kommt in unterschiedlichen Varianten vor, u.a. als ↑Test gegen eine Spezifikation (vgl. ↑Unittest). Gegenstück: ↑statische Codeanalyse.

Feature Request Anfrage nach einer neuen Funktion einer Software, die oft über einen ↑Issue Tracker an die Entwickler gestellt wird. Die Verwendung eines ↑Issue Trackers oder einer ↑Bug-Verwaltung für diesen Zweck erlaubt die Strukturierung der Kommunikation zwischen Anfragendem und Entwicklern, die Archivierung einer solchen Anfrage und die Zuweisung zu einem konkreten Entwickler zur Bearbeitung.

Gödelscher Unvollständigkeitssatz einer der wichtigsten Sätze der modernen Logik. Befasst sich mit der Ableitbarkeit von Aussagen in formalen Systemen und zeigt die Grenzen der formalen Systeme ab einer bestimmten Leistungsfähigkeit auf. Nachweis, dass es in hinreichend starken Systemen, wie der Arithmetik, Aussagen geben muss, die man weder formal beweisen noch widerlegen kann. Wurde 1931 von Kurt Gödel veröffentlicht und bedeutete das Ende des Hilbertprogramms, das die Widerspruchsfreiheit der Mathematik ausgehend von Axiomen zu beweisen suchte.

Halteproblem Fragestellung aus der theoretischen Informatik. Beschreibt die Frage, ob die Ausführung einer Rechenvorschrift (Algorithmus) zu einem Ende gelangt. Alan Turing konnte beweisen, dass es keinen Algorithmus gibt, der diese Frage für alle möglichen Algorithmen und beliebige Eingaben beantwortet. Das Halteproblem ist somit algorithmisch nicht entscheidbar. Dieses Ergebnis ist von grundlegender Bedeutung für die Berechenbarkeitstheorie (↑Berechenbarkeit). Die generelle Unentscheidbarkeit des Halteproblems ist eng verwandt mit ↑Gödels Unvollständigkeitssatz.

Hotfix Kurzfristige Behebung eines (den Entwicklern) bekannt gewordenen kritischen Fehlers in einer Produktivversion der Software. Wird in einer Produktivversion ein Fehler bekannt, der zügig behoben werden soll, ohne auf die nächste Produktivversion zu warten, bietet sich die Behebung in einem zugehörigen kurzlebigen Branch an, der direkt von der letzten Produktivversion abzweigt, den Fehler behebt, und anschließend wieder mit dem Hauptzweig zusammengeführt wird (*merge*). Wichtig ist dabei, die Fehlerbehebung auch in den Entwicklungszweig einfließen zu lassen – und ggf. entsprechende ↑Tests zu entwickeln, die ihn automatisch detektieren können.

Issue Tracker Software zum Empfang, zur Bestätigung, Klassifizierung und Bearbeitung von Kundenanfragen. Erklärtes Ziel ist, dem Verlust von Nachrichten vorzubeugen und jederzeit einen Gesamtüberblick über die zu bearbeitenden Vorgänge zu ermöglichen. Ein Issue Tracker erlaubt die Zuweisung einer Anfrage (*ticket*) an einen Verantwortlichen zur weiteren Bearbeitung bis zur Lösung. Die Grenzen zur ↑Bug-Verwaltung sind mitunter fließend, andererseits bilden Issue Tracker mitunter einen integralen Bestandteil einer Projektmanagement-Software.

Korrektheit Eigenschaft eines Computerprogramms, einer Spezifikation zu genügen (vgl. ↑Verifizierung)

Pair Programming „Paarprogrammierung“, enge und aktive Zusammenarbeit zweier Programmierer bei der Programmierung von Software. Eine Person sitzt an der Tastatur und pro-

grammiert, während die andere aktiv über den entstehenden Code und das Problem nachdenkt und ggf. korrigierend eingreift. Auffallende Probleme werden sofort angesprochen und diskutiert. Die Rollen sollten häufig wechseln (mindestens innerhalb einer Stunde), idealerweise ebenso die Zusammensetzung der Paare.

Patch (engl. für „Flicken"), aktualisierte Version einer Software, die meist ausschließlich der Fehlerbehebung dient.

pull Holen der Historie aus einem anderen (ggf. entfernten) ↑Repository; relevant bei dezentralen Systemen zur ↑Versionsverwaltung.

Pull Request Bei der Verwendung eines dezentralen Systems zur ↑Versionsverwaltung die Anfrage an die (Haupt-)Entwickler, eine bestimmte Änderung aus einem anderen ↑Repository in ihren Entwicklungszweig zu übernehmen.

Regression erneutes Auftreten eines einmal behobenen Fehlers in einer neuen Version. Lässt sich grundsätzlich dadurch umgehen, dass einmal detektierte Fehler in ↑Tests, meist ↑Unittests, verwandelt werden und jede zukünftige Version einer Software entsprechend (automatisch) auf das Auftreten dieses Fehlers überprüft wird (↑Testsuiten).

Repository Versionsdatenbank, (zentraler) Speicherort der versionierten Dateien

statische Codeanalyse ↑Codeanalyse ohne Ausführung des Quellcodes. Kann neben Syntaxfehlern und Einhaltung von Konventionen mitunter auch komplexere Tests durchführen. Häufig in einer ↑IDE integriert, so dass bereits beim Schreiben des Quellcodes auf mögliche Probleme hingewiesen wird. Gegenstück: ↑dynamische Codeanalyse.

Test hier: strukturiertes Vorgehen, eine Software zu überprüfen. Setzt die Definition klarer Anfangs- und Endbedingungen (Eingabe und Ergebnis) voraus und sollte idealerweise vollständig automatisiert ablaufen können. Vgl. ↑dynamische Codenanalyse und ↑Unittest.

Testsuiten Sammlungen von Testroutinen, die in einem funktionalen Zusammenhang stehen, z.B. einen bestimmten Teil eines Programmes abtesten. Lassen sich meist automatisch ausführen. Bestehen oft aus ↑Unittests.

Typisierung (*typing*) Zuweisung eines Typs zu einem Objekt (im abstrakten Sinne) einer Programmiersprache, z.B. Ganzzahl (*integer*) oder Zeichenkette (*string*) im Fall einer Variable. Abstraktion, die die Ausdrucksstärke von Programmiersprachen und Programmen deutlich erhöht, und die Überprüfung der Korrektheit erleichtert sowie Optimierungen ermöglicht. Typisierung kann explizit und implizit erfolgen. Darüber hinaus wird zwischen starker und schwacher Typisierung sowie zwischen statischer und dynamischer Typisierung unterschieden. Jede Art der Typisierung hat ihre Vor- und Nachteile, und unterschiedliche Programmiersprachen verwenden unterschiedliche Arten der Typisierung.

Unittest ↑Test eines Codeblocks in Isolation. Ein Unittest überprüft von außen, ohne den Quellcode des zu testenden Systems zu kennen oder zu benötigen. Die getesteten Codeblöcke sind i.d.R. klein. Zwingende Voraussetzung ist, dass das (erwünschte) Verhalten des zu testenden Codeblocks eindeutig definierbar (und seinerseits in Form von Quellcode formalisierbar) ist. Unittests sind gewissermaßen die unterste Ebene (automatisierter) ↑Tests.

Validierung hier: Prüfung der Eignung beziehungsweise des Wertes einer Software bezogen auf ihren Einsatzzweck auf Grundlage eines vorher aufgestellten Anforderungsprofils

Verifizierung hier: Feststellung, ob ein Computerprogramm seiner Spezifikation entspricht, und Aufspüren vorhandener Fehler

Versionsverwaltung engl. *version control system*, VCS; Software zur Verwaltung unterschiedlicher Versionen von Dateien und Programmen, die den Zugriff auf beliebige ältere als Versionen (Revision) gespeicherte Zustände ermöglicht. Gleichzeitig ein wichtiges Werkzeug für die Softwareentwicklung und wesentlicher Aspekt einer Projektinfrastruktur.