Softwareentwicklung in der Wissenschaft

Betrachtung der Entwicklungsqualität des Planet Simulator

Enno Köster

30. März 2011

Inhaltsverzeichnis

1 Einleitung

Im Rahmen des Seminars "Softwareentwicklung in der Wissenschaft" soll untersucht werden, inwiefern die theoretischen Grundlagen der Entwicklung hochqualitativer Software bei der Entwicklung wissenschaftlicher Software Anwendung finden. Falls es größere Diskrepanzen zwischen Theorie und Praxis gibt, ist herauszuarbeiten worin die Gründe für das Abweichen von den durch das Software-Engeneering gegebenen Best Practises liegt.

Die Grundlage für diese Untersuchung bildeten einige Vorträge, in denen wichtige Punkte des Software-Engineerings herausgearbeitet wurden. Den Anfang machte ein allgemeiner Einführungsvortrag, welcher einen guten Überblick über das gesamte Thema gab. Darauf folgten Vorträge zu den Themen Versionierung, Code-Qualität, Softwaretests, Release-Management und der Genauigkeit von Berechnungen. Diese gaben jeweils die Best Practises in den einzelnen Bereichen wider, um eine einheitliche theoretische Basis bei allen Teilnehmern zu schaffen, mit der die zu untersuchenden Projekte verglichen werden konnten.
Auf Basis dieser Vorträge wurden in verschiedenen Projekten Interviews durchgeführt, um einen Einblick in die Paxis wissenschaftlicher Softwareentwicklung zu bekommen.

Diese Ausarbeitung beschäftigt sich mit der Entwicklung des Planet Simulators [2], einer Software, die der Simulation und Erforschung der Atmosphäre dient. Im folgenden sollen die in einem Interview mit dem Hauptentwickler gesammelten Erkenntnisse über die Praxis in seinem Projekt mit der Theorie zum Thema Software-Qualität verglichen werden und untersucht werden, wie es um die Entwicklung und die Qualität des Planet Simulators bestellt ist.

2 Vorbemerkungen

2.1 Methodischer Aufbau des Dokuments

Basis dieser Arbeit ist der internationale Standard ISO/IEC 9126-1, welcher Charakteristika definiert, anhand derer die Qualität einer Software bewertet werden kann. Im folgenden werden alle relevanten Punkte des Standards kurz erklärt und im Bezug auf die zu bewertende Software betrachtet. Die Untersuchung der Einzelaspekte ist dabei sehr unterschiedlich detailiert und spiegelt in etwa ihre Gewichtung im Interview mit dem Entwickler wider.
Alle Informationen über das Projekt, welche im Folgenden erwähnt werden, stammen aus zwei mit dem Hauptentwickler Dr. Edilbert Kirk geführten Interviews. Das erste Interview war noch sehr allgemein gehalten, während zum Termin des zweiten Interviews klar war, dass eine Bewertung anhand des oben genannten ISO-Standards die sinnvollste Herangehensweise sein würde und die Fragen spezifischer waren.
Ziel dieser Arbeit ist die Untersuchung der Softwareentwicklungsqualität des Planet Simulators. Die Einschätzung der Qualität ist ein in vielen Aspekten sehr subjektiver Vorgang, der oft stark von der Meinung der zum Produkt befragten Person beeinflusst wird. Die strukturelle Nähe der Arbeit zum ISO-Standard soll gewähleisten, dass kein wichtiger Aspekt unbeachtet bleibt. Der Author hat versucht, die Subjektivität der Antworten im Interview durch geeignete Nachfragen zu entdecken und im Zuge der Erstellung der Ausarbeitung zu beachten und zu normalisieren.

2.2 Interviewpartner und Institut

Der Interviewpartner, von dem die Aussagen bezüglich des Planet Simulators stammen, ist Dr. Edilbert Kirk, seit über 25 Jahren Wissenschaftler an der Universität Hamburg. Seit 2000 liegt sein Schwerpunkt auf der Entwicklung des Planet Simulators, welcher zur Zeit der Erstellung der Ausarbeitung im Rahmen des Klima-Exzellenzclusters CliSAP in Hamburg weiterentwickelt wird.
Der Exzellenzcluster CliSAP ist ein von Bund und Ländern geförderter Zusammenschluss der Universität Hamburg, dem Max-Planck-Institut für Meteorologie, dem Deutschen Klimarechenzentrum sowie dem Helmholtz-Zentrum Geesthacht. In ihm sind Meteorologen, Meereskundler, Geophysiker und Ökologen mit Sozial- und Wirtschaftsexperten, aber auch mit Medienwissenschaftlern und Friedensforschern vereint.

2.3 Der Aspekt der Konformität

Ein Charakteristikum, das in allen Bereichen des Standards als Unterpunkt vorhanden ist, die Konformität, soll nur hier im Vorfeld kurz betrachtet werden:
Die Entwickler des Planet Simulators streben in keiner Weise die Einhaltung von außen definierter Standards und Normen an. Es gibt lediglich interne Vorgaben zur Schreibweise des Quelltexts und Codekommentaren, welche die Lesbarkeit und später eventuell erforderliche Änderbarkeit des Codes gewährleisten sollen und strikt befolgt werden.
Der Großteil vorhandener Normen würde nach Meinung des Authors ohnehin gar nicht oder nur ungenügend auf diese Software anwendbar sein. Die Aufgaben von Software in der Forschung unterscheiden sich teils sehr stark von den Aufgaben und Anwendungsbereichen üblicher Desktopsoftware, für welche viele Standards geschrieben werden.
Einzig die Einhaltung gewisser Normen bezüglich der GUI wäre vielleicht erstrebenswert, um z.B. die Kompatibilität mit Screenreadern oder allgemein die Benutzerfreundlichkeit zu fördern. Allerdings gilt nach Einschätzung des Authors auch in diesem Bereich, dass der größte Teil der von GUI-Standards gemachten Vorgaben auch durch gesunden Menschenverstand abgedeckt werden.

3 Funktionalität

Die folgenden Punkte betreffen die Frage, inwiefern die Software die Funktionen bereitstellt, welche durch ihr Anwendungsgebiet und ihre Anwender gefordert werden [1].

3.1 Angemessenheit

Erfüllt die Software die spezifizierten Aufgaben durch entsprechende (Unter-)Funktionen?
Anders herum formuliert: Erledigt die Software bei Benutzung auch Dinge, welche nicht zwingend erforderlich oder sogar kontraproduktiv sind, oder kümmert sie sich nur um notwendige Vorgänge?
Der Planet Simulator hat durch Startskripte und ebenfalls durch eine GUI, in der jeder Aspekt konfiguriert werden kann, die Möglichkeit klar zu definieren, was zu tun ist. An diese klaren Definitionen hält sich der Simulationsteil des Programms. Der modulare Aufbau der Software ermöglicht das selektive Laden aller Teile, welche nicht zwingend für die Funktion des Kerns notwendig sind. Das Laden und Entladen von Modulen ist sogar während des Simulationslaufs möglich.
Die Software ist also nach Definition von ISO/IEC 9126 angemessen in ihrer Funktionalität.

3.2 Richtigkeit

Liefert die Software die richtigen oder vereinbarten Ergebnisse oder Wirkungen, zum Beispiel die benötigte Genauigkeit von berechneten Werten [3]?
Diese Frage erfordert eine Antwort in zwei Teilen, da es einen Bereich gibt, in dem diese Frage klar zu beantworten ist, und einen anderen, in dem die Lage nicht mit vollständiger Sicherheit zu beantworten ist. Die angestellten Berechnungen von Fourier-Transformationen und anderen verwendeten mathematischen Verfahren können einfach mit bekannt korrekten Werten abgeglichen werden, sodass hier ganz klar zu sagen ist, dass die Ergebnisse richtig sind. Es wird bei Optimierungen z.B. an oben genannten Fourier-Transformationen, darauf geachtet, dass sie hinterher auch exakt das gleiche ergeben. Außerdem wird für den Checkpoint-Mechanismus, welcher z.B. einmal pro simuliertem Jahr alle Daten speichern kann, gewährleistet, dass er das Ergebnis nicht verfälscht. Das heisst, das Ergebnis eines Laufs über 10 Jahre ist exakt das gleiche, wie das von 10 aufeinander folgenden Läufen die jeweils als Eingabe das gespeicherte Ergebnis des Vorgängers nutzen. Allgemein gilt bei technischen Änderungen, dass das gleiche Ergebnis produziert werden soll. Die Richtigkeit in Bezug auf die mathematischen Verfahren und die Ein-/Ausgabe ist also gesichert.
Die Frage, ob die physikalischen Formeln korrekt umgesetzt wurden, ist etwas schwieriger zu beantworten. Hier ist der meteorologische Sachverstand des Implementierenden gefragt. Oft ist auch hier ein Vergleich mit bekannt korrekten Werten sinnvoll. Diese Software versucht jeden einzelnen simulierten Aspekt für sich physikalisch korrekt zu reproduzieren.
Die Zielsetzung der Software ist eine andere als z.B. ein Klimamodell wie ECHAM, das für IPCC-Reports über das zukünftige Weltklima herangezogen werden. Ein Abgleich mit realen Wetterwerten aus der Vergangenheit kann also nicht genau die richtigen Werte simulieren, bis jeder atmospärische Effekt korrekt simuliert wird. Auch ein Vergleich mit Vorgängerversionen ist aus ähnlichen Gründen problematisch, wenn neue Module die Simulation vielleicht beeinflussen.
Wenn man Richtigkeit also als die korrekte Berechnung der einzelnen physikalischen Komponenten definiert, ist diese gegeben. Wenn man eine realitätstreue Simulation des aktuellen Wetters als richtig ansieht und damit die Zielsetzung des Projekts ignoriert, ist das berechnete Ergebnis noch nicht ganz exakt. Dieser Umstand ist allerdings auch der beste Grund für die Weiterentwicklung des Programms.

3.3 Interoperabilität

Fähigkeit, mit vorgegebenen Systemen zusammenzuwirken [3]. Der Planet Simulator hat, wie alle anderen Klimamodelle, keine Netzwerkschnittstelle mit der man wie z.B. mit einem Webserver kommunizieren kann, sodass sich die Frage nach der Zusammenarbeit mit anderer Software auf den Import und Expoprt von Datenformaten beschränkt. Beides ist sowohl mit dem in der Meteorologie weit verbreiteten NetCDF und GRIB als auch Service, einem einfacheren Format, möglich, sodass man der Software eine gute Interoperabilität attestieren kann.
Das Modell braucht für seine grundlegende Funktion allerdings viel weniger Daten als diese Formate enthalten können. Einige Randdaten (Topologie, Oberflächentemperatur, Albedo, Solarkonstante,etc.) reichen für eine Simulation schon aus.

3.4 Sicherheit

Fähigkeit, unberechtigten Zugriff, sowohl versehentlich als auch vorsätzlich, auf Programme und Daten zu verhindern. [3]
Dieser Aspekt ist für diese Art von Software wenig relevant, da keine von außen zugängliche Netzwerkschnittstelle existiert und das Format für Eingabedaten klar definiert ist und bei Nichteinhaltung das Programm seinen Dienst einstellt. Wenn bei der Überprüfung des Formats auch in Zukunft an alle notwendigen Überprüfungen gedacht wird, sollte es absolut keine Angriffsmöglichkeit gegen das Programm geben.

3.5 Ordnungsmäßigkeit

Merkmale von Software, die bewirken, dass die Software anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen und ähnliche Vorschriften erfüllt. [3]
Normen oder gesetzliche Bestimmungen existieren für das Anwendungsgebiet Klimamodelle nicht, sodass dieser Punkt den Planet Simulator nicht relevant ist.
Wie schon bei der Richtigkeit betont wurde, ist das Ziel jeden physikalischen Effekt korrekt zu simulieren. Systematische Fehler in Einzelbereichen werden akzeptiert, wenn diese noch nicht ausreichend erforscht sind. Dies steht im krassen Gegensatz zu Modellen, welche zur Wetterberechnung oder zu Aussagen über das zukünftige Klima herangezogen werden, welche versuchen solche Fehler durch teils komplizierte Maßnahmen zu korrigieren. Dies ist für die unmittelbare Vorhersage sinnvoll, widerspricht allerdings dem Ziel des Planet Simulators, die Atmoshäre vollständig korrekt zu simulieren.

4 Zuverlässigkeit

Kann die Software ein bestimmtes Leistungsniveau unter bestimmten Bedingungen über einen bestimmten Zeitraum aufrechterhalten [3]? - Leidet die Software unter häufigen Abstürzen? Wie geht sie mit gegebenenfalls entstandenen Problemen um? Bricht sie ab, pausiert sie, oder ignoriert sie das Problem? Wieviel Aufwand ist nötig, um Fehler zu behandeln bzw. den Vorgang zu wiederholen?

4.1 Reife

Geringe Versagenshäufigkeit durch Fehlerzustände. [3]
Der Software kann man eine sehr hohe Reife attestieren, welche durch die lange Entwicklungsgeschichte und die starke Beachtung von Code-Richtlinien begründet ist. Es wurden schon Läufe über mehrere 100.000 Jahre Fehlerfrei und ohne Unterbrechnung erfolgreich durchgeführt.

4.2 Fehlertoleranz

Fähigkeit, ein spezifiziertes Leistungsniveau bei Software-Fehlern oder Nicht-Einhaltung ihrer spezifizierten Schnittstelle zu bewahren. [3]
Beim Start wird eine Überprüfung der Eingabewerte durchgeführt, welche sicherstellt, dass keine pysikalisch unmöglichen Daten verwendet werden. So werden z.B. Temperaturen unter dem absoluten Nullpunkt von null Kelvin als nicht zulässig abgelehnt und der Lauf nicht ausgeführt.
An vielen anderen Stellen sind allerdings extreme Werte, wie z.B. sehr starke Winde erlaubt, da der Planet Simulator gerade dem Zweck dient, zu sehen, wie sich Änderungen gewisser Parameter der Atmosphäre auswirken. Wenn die Eingaben zu übertrieben gewählt werden, kann dies zum Absturz der Software führen. Allerdings sind dies üblicherweise Fälle, die eher der Neugier des Anwenders bezüglich des Verhaltens der Software entspringen als wissenschaftlich sinnvolle Simulationen.
Die Fehlertoleranz ist also auf einem absolut akzeptablen Niveau, denn mehr Überprüfungen von Wertebereichen würden eventuell den interessierten Wissenschaftler einschränken.

4.3 Wiederherstellbarkeit

Fähigkeit, bei einem Versagen das Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen. Zu berücksichtigen sind die dafür benötigte Zeit und der benötigte Aufwand. [3]
Es gibt die Möglichkeit nach selbst bestimmten Zeitintervallen Zwischenergebnisse der Berechnungen auf einen persistenten Speicher(Festplatte o.Ä.) zu schreiben. Bei großen Läufen wird dieser Checkpointing-Mechanismus üblicherweise einmal pro simuliertem Jahr genutzt. Bei einem Systemabsturz kann von diesem Punkt an die Berechnung wieder fortgesetzt werden. Die Nutzung dieses Sicherungssystems ist unkompliziert.

5 Benutzbarkeit

Welchen Aufwand fordert der Einsatz der Software von den Benutzern und wie wird er von diesen beurteilt? Aufwand, der zur Benutzung erforderlich ist, und individuelle Beurteilung der Benutzung durch eine festgelegte oder vorausgesetzte Benutzergruppe [3].
Die Benutzergruppe soll im folgenden aus Meteorologen mit fortgeschrittenem oder abgeschlossenem Studium bestehen, welche entsprechendes Wissen z.B. über Fachbegriffe hat.

5.1 Verständlichkeit

Aufwand für den Benutzer, das Konzept und die Anwendung zu verstehen [3] und herauszufinden, ob und wie die Software für seine Zwecke verwendbar ist [1].
Ob der Planet Simulator das richtige Werkzeug ist, sollte mit Hilfe der Dokumentation und ein paar eigenen Versuchen unkompliziert herauszufinden sein, denn die Software ist durch ihre GUI relativ einfach Bedienbar, mehr dazu jedoch im nächsten Abschnitt.

5.2 Erlernbarkeit

Wie Aufwändig ist das erlernen der Bedienung der Software? Der Planet Simulator ist definitiv nur für fortgeschrittene Nutzer geeignet, um produktiv Simulationen durchzuführen. Die Bedienung ist zwar durch die GUI möglichst einfach und Einsteigerfreundlich gehalten, aber die oben gemachte Einschränkung auf Meteorologen ist aufgrund der Fachbegriffe, welche nicht sinnvoll zu ersetzen sind, notwendig. Eine Simulation zu starten ist einfach, sinnvolle Parameter für die Randdaten zu wählen jedoch nicht. Ein fundiertes Wissen in den Bereichen theoretische Meteorologie und Numerik ist empfohlen. Die Erlernbarkeit zu beurteilen fällt durch fehlende Kenntnisse des Authors schwer, wird jedoch im Vergleich zu anderen Modellen als besser angesehen, da durch die optionale Unterstützung des Startvorgangs durch eine GUI die Erstbenutzung erleichtert wird.

5.3 Bedienbarkeit

Aufwand für den Benutzer, die Anwendung zu bedienen [3].
Wie im vorhergehenden Abschnitt erläutert, ist die Bedienung bei vorhandenem Know How im meteorologischen Bereich einfach zu erlernen. Ein weiterer Punkt ist die Frage, wie aufwendig es ist, Parameter der Simulation zu ändern. Dies wird ebenfalls unkompliziert von der GUI unterstützt, sodass eine Veränderung von Konfigurationsdateien und Startskripten zwar grundsätzlich für den versierten Nutzer möglich, aber nicht zwingend nötig ist. Es ist sogar möglich Parameter während der laufenden Simulation zu ändern und die Effekte dieser Veränderung sofort zu sehen. Dies wird durch die optionale Kopplung des Kerns der Simulation an die GUI ermöglicht. Ein Beispiel für eine änderbare Größe wäre die Solarkonstante, die angibt, wieviel Energie von der Sonne auf den simulierten Planeten fällt, oder die CO_2-Konzentration eines Gitterpunktes. Technisch möglich wäre die Änderung jedes beliebigen Parameters, allerdings ist diese Funktionalität im Moment nicht für alle Parameter implementiert, da ihre Änderung von Hand teilweise unsinnig wäre und den Aufwand nicht rechtfertigt.

5.4 Attraktivität

Wie attraktiv ist die Software für einen Benutzer und was macht sie interessant? Das zentrale Argument für die Verwendung des Planet Simulators ist die GUI, welche die Benutzung der Software sehr viel einfacher macht als andere Modelle, welche zwingend voraussetzen, dass man Startskripte editiert und nach einem Lauf die Visualisierung externen Programmen überlassen. Sowohl der Startvorgang als auch die Visualisierung der Daten werden von der GUI unterstützt.
Was die meteorologische Seite angeht, geht die Attraktivität für dieses Modell hauptsächlich vom Schwerpunkt auf der Atmosphäre aus. Andere Modelle gewichten ihren Aufwand anders und legen ihren Fokus z.B. auf die Simulation der Ozeane.

6 Effizienz

Wie liegt das Verhältnis zwischen Leistungsniveau der Software und eingesetzten Betriebsmitteln?
Verhältnis zwischen dem Leistungsniveau der Software und dem Umfang der eingesetzten Betriebsmittel unter festgelegten Bedingungen. [3].

6.1 Zeitverhalten

Antwort- und Verarbeitungszeiten sowie Durchsatz bei der Funktionsausführung [3].
Der Planet Simulator ermöglich den Onlinebetriebt, bei dem der Verlauf der Simulation direkt visuell dargestellt wird. Die Antwortzeit der GUI ist sehr gut und es gibt praktisch keine Verzögerung. Eingaben werden sofort angenommen. Änderungen an Simulationsparametern in der GUI werden sofort nach Bestätigung im Simulationsteil aktiv.

6.2 Verbrauchsverhalten

Anzahl und Dauer der benötigten Betriebsmittel bei der Erfüllung der Funktionen. Ressourcenverbrauch, wie CPU-Zeit, Festplattenzugriffe usw. [3]
Im Vergleich mit anderen Modellen ist das Modell durch Jahrzehnte lange Optimierung sowohl im Bezug auf Speicherverbrauch als auch im Bedarf an CPU-Zeit sehr sparsam. Außerdem ist es durch die anpassbare Auflösung der Gitterpunkte sehr flexibel einsetzbar. Grundsätzlich wäre es mit einer niedrig gewählten Auflösung durchaus möglich auf einem leistungsschwachen Netbook eine Simulation durchzuführen. Diese wäre wahrscheinlich nicht so präzise wie ein Lauf mit höherer Auflösung auf einem starken Workstation-Rechner, aber Aussagen über den Trend gewisser Größen könnten dennoch ohne Probleme gemacht werden.

7 Wartbarkeit

Welchen Aufwand erfordert die Durchführung vorgegebener Änderungen an der Software? - Aufwand, der zur Durchführung vorgegebener Änderungen notwendig ist. Änderungen können Korrekturen, Verbesserungen oder Anpassungen an Änderungen der Umgebung, der Anforderungen oder der funktionalen Spezifikationen einschließen [3].

7.1 Analysierbarkeit

Wie hoch ist der Aufwand, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen [3]. Auf die Lesbarkeit des Quelltextes wird sehr streng geachtet, um das Debugging so einfach wie möglich zu halten. Es gibt Regeln zur Formatierung Codes, um ein einheitliches Bild zu haben. Des weiteren wird darauf geachtet, dass der komplette Quelltext gut kommentiert ist, sodass das Nachvollziehen einzelner Passagen des Codes im Fehlerfall zügig und einfach zu erledigen ist.
Optional aktivierbare Debug-Ausgaben begünstigen ebenfalls das finden von Fehlern. Der modulare Aufbau des gesamten Programms unterstützt die Korrektur von Bugs weiter, da die Ursache eines Fehlers üblicherweise nur an einer Stelle zu finden ist.
Generell wird viel Wert auf die Debugging-Fähigkeit des Codes gelegt.

7.2 Modifizierbarkeit

Aufwand zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder Anpassung an Umgebungsänderungen [3].
Der im vorhergehenden Abschnitt schon erwähnte modulare Aufbau und die strikten Regeln zur Formatierung und Kommentierung begünstigen auch die Modifizierbarkeit des Quelltexts. Wenn ein Fehler auftritt, ist dieser normalerweise schnell korrigiert und kleinere Änderungen sind einfach zu implementieren. Bei größeren Änderungen an einem Modul wird empfohlen gleich ein neues auf Basis des alten zu schreiben, unter anderem um beide Versionen zu vergleichen.

7.3 Stabilität

Wahrscheinlichkeit des Auftretens unerwarteter Wirkungen im Zuge von Änderungen an der Software [3].
Bei Entwicklung eines neuen Moduls ist das auftreten Softwaretechnisch unerwarteter Wirkungen durch die klar definierten Schnittstellen für Module sehr unwahrscheinlich. Unerwartet ist höchstens das meteorologische Ergebnis, welches bei hinzufügen eines neuen physikalischen Effekts zur Simulation manchmal schwer vorauszusagen ist. Am Ende sollte das Modell jedoch realistischer werden.

7.4 Testbarkeit

Aufwand, der zur Prüfung der geänderten Software notwendig ist [3].
Bei technischen Änderungen wie der Optimierung von Fourier-Transformationen ist der Aufwand sehr gering, da nur getestet werden muss, ob die neue Version das gleiche berechnet wie die alte. Allgemein ist die Testbarkeit der Berechnungen eines Moduls bei Änderungen recht gut.
Bei Hinzufügen eines Moduls, also inhaltlichen Änderungen am Programm kann das Testen auf Korrektheit allerdings beliebig schwierig werden, da oft nicht genau bekannt ist, wie das Ergebnis aussehen muss. Die Diagnose solcher Änderungen wird sehr durch die GUI unterstützt, die in der Lage ist viele Größen zu visualisieren. Die Korrektheit des neuen Moduls wird mit den Ausgaben des Programms und meteorologischem Sachverstand untersucht.

8 Portabilität

Wie leicht lässt sich die Software in eine andere Umgebung übertragen? Eignung der Software, von der Umgebung in eine andere übertragen werden zu können. Umgebung kann organisatorische Umgebung, Hardware- oder Software-Umgebung sein [3].

8.1 Anpassbarkeit

Fähigkeit der Software, diese an verschiedene Umgebungen anzupassen. Grundsätzlich läuft der Kern der Simulation auf allen Plattformen, für die ein Fortran-Compiler existiert, da der Kern komplett plattformunabhängig geschrieben wurde. Für die Ausführung ist außerdem die tcsh-Shell erforderlich, welche allerdings auch für die meisten Plattformen verfügbar ist.
Für die GUI ist außerdem die Entwicklungsbibliothek für X.org, eine grafische Benutzeroberfläche, notwendig. Die beiden Abhängigkeiten machen es unmöglich die Software direkt auf Windows auszuführen. Allerdings kann man sich hier behelfen, indem man eine Cygwin-Umgebung(Emulation einer Linux-Umgebung) oder eine virtuelle Maschine mit passendem Betriebssystem nutzt. Die Performance der Software leidet nur wenig darunter und ist nur geringfügig langsamer als bei nativer Ausführung.

8.2 Installierbarkeit

Aufwand, der zum Installieren der Software in einer festgelegten Umgebung notwendig ist [3].
Wenn die Plattform Linux oder eine andere unixoide Plattform ist, muss lediglich, falls nicht schon vorhanden, ein Fortran-Compiler und die tch-Shell installiert werden und schon ist die Software lauffähig. Die Nutzung der GUI setzt außerdem einen funktionierenden X-Server und das xorg-dev Paket voraus. Eine weitere Konfiguration der Software ist in Normalfall nicht notwendig, da ein umfangreiches automatisches Configure-Skript, alle eventualitäten abdeckt. Somit ist die Installation auf einem Linux-Rechner unkompliziert.
Auf einem Windows-Rechner müsste man wie oben in Anpassbarkeit erwähnt eine virtuelle Maschine mit einer Linux-Distribution installieren, um die Software nutzen zu können. Damit ist die Installation unter Windows mit einem erheblichen Mehraufwand verbunden.

8.3 Koexistenz

Fähigkeit der Software neben einer anderen mit ähnlichen oder gleichen Funktionen zu arbeiten [3].
Dieser Punkt der ISO-Norm ist für die hier behandelte Software absolut nicht relevant, da Klimamodelle abgesehen von CPU und Speicher keine gemeinsamen Ressourcen nutzen bei denen es zu Konflikten kommen könnte.

8.4 Austauschbarkeit

Möglichkeit, diese Software anstelle einer spezifizierten anderen in der Umgebung jener Software zu verwenden, sowie der dafür notwendige Aufwand [3].
Es gibt keinen Standard und keine Protokolle für Klimamodelle an die man sich halten könnte. Demnach ist es schwierig bis unmöglich sie untereinander auszutauschen. Allein die Start-Skripte und Parameter sind schon so unterschiedlich, dass man sie für ein anderes Modell komplett neu schreiben muss. Dies liegt unter anderem am Unterschiedlich gelegten Schwerpunkt der Modelle. Der kleinste gemeinsame Nenner wäre wahrscheinlich NetCDF, ein Format für Eingabgedaten, das die meisten Klimamodelle unterstützen.

9 Zukunftsaussichten für das Projekt

Der Planet Simulator blickt nach Meinung des Authors in eine sehr positive und lange Zukunft. Dies hat mehrere im Folgenden erläuterte Gründe:
Es gibt zur Zeit der Erstellung dieses Dokuments Bestrebungen das Projekt von der fast ausschließlich im Akademischen Umfeld zum wissenschaftlichen Rechnen genutzten Sprache FORTRAN auf die sehr viel weiter verbreitete Sprache C++ zu portieren. Dies würde die Software einer sehr viel breiteren Masse an Entwicklern öffnen. Die Weiterentwicklung der Software könnte dadurch zusätzlich an Fahrt gewinnen und in Folge dessen die Attraktivität der Software weiter steigern.
Ein weiterer Punkt der von der ISO/IEC-9126-1 nicht erfasst wird, und dessen Wichtigkeit leider in vielen Dokumenten unterschätzt wird, ist der vorbildliche Support, welcher im Anwender-Forum auf der Projektwebseite durch die Entwickler geleistet wird. Die hier geleistete Hilfe ist nicht selbstverständlich und soll daher Erwähnung finden.
Außerdem spricht für das Projekt, dass Methoden der Softwareentwicklung als unterstützend wahrgenommen werden, was in den Augen des Authors die Wahrscheinlichkeit einer anhaltend hohen Qualität der Software höher erscheinen lässt als in ähnlichen ihm bekannten Projekten.

10 Fazit

Die im Zuge des Seminars geführten Interviews waren sehr interessant und haben einen guten Eindruck sowohl von der Planet Simulator Software und ihrer Entwicklung als auch vom Interviewpartner Dr. Edilbert Kirk hinterlassen.
Das Interview zeichnete sich durch eine offene und lockere Atmosphäre aus. Es wurde keiner Frage ausgewichen und alle Unklarheiten wurden bei entstehen umgehend wieder beseitigt. Das Interview hat einen guten Einblick in die Praxis der Entwicklung am Planet Simulator gewährt.
Die Softwareentwicklung ist in diesem Projekt nicht nur ein notwendiges Übel, sondern wird als wichtiges Element der Forschung angesehen, ohne das vieles nicht möglich wäre. Es wird bei der Entwicklung sehr darauf geachtet sauber zu arbeiten und ordentlichen aufgeräumten Code zu produzieren. Der Ursprung dieser Software liegt schon mehr als 30 Jahre zurück, doch im Gegensatz zu manchen anderen Projekten von denen in den Vorträgen zu hören war, lebt man hier nicht mit Altlasten und unsauberem, schwer zu wartendem Code, sondern überarbeitet problematische Teile, um eine anhaltend hohe Qualität der Software zu gewährleisten.
Der modulare Aufbau des Quelltexts erleichtert die Arbeit an der Software sehr, sodass jegliche notwendige Anpassungen leicht von der Hand gehen. Die klar definierte Schnittstelle für die einzelnen Module der Simulation erleichtert die Erweiterung der Software um neue Aspekte.
Einzig die GUI, ein besonderes Merkmal, das den Planet Simulator von der Bedienerfreundlichkeit her über andere Modelle stellt, könnte noch ein wenig Feinschliff vertragen. Die Oberfläche ist aufgeräumt und bietet alle Möglichkeiten die ein Anwender sich wünschen könnte, aber sie könnte den Einstieg noch durch eine Hilfefunktion erleichtern, die zu jedem Parameter auf wenigen Zeilen grob umreisst, was er Bedeutet.
Insgesamt kann man sagen, dass der Planet Simulator eine ausgezeichnete Software ist, die von engagierten Entwicklern betreut wird. Die Qualität der Software ist in diesem Projekt ein zentraler Punkt, dem viel Beachtung geschenkt wird.
Was den Vergleich der Praxis mit der Theorie selbst anbelangt, zeigt sich, dass selbst ein sehr generisch gehaltener Standard wie der ISO/IEC-9126, nicht perfekt passend für einen Vergleich mit wissenschaftlicher Software ist, da Charakteristika im Standard existieren die nicht relevant sind. Man muss also, egal welches bekannte Maß aus der Softwareentwicklung man heranzieht, die Anforderungen ein wenig dem speziellen Gebiet anpassen, um der Software gerecht zu werden.

Literatur

[1] ISO/IEC. *ISO/IEC 9126. Software engineering – Product quality*. ISO/IEC, 2001.

[2] Meteorologisches Institut Universität Hamburg. Planet simulator, 2011. $http://www.mi.uni-hamburg.de/index.php?id=216$.

[3] Wikipedia. Iso/iec 9126, 2011. $http://de.wikipedia.org/wiki/ISO/IEC\_9126$.