FVA-Richtlinie für die Beantragung und Durchführung von Softwareentwicklungsprojekten

Forschungsvereinigung Antriebstechnik e.V.

(Kurztitel: FVA-Softwareentwicklungsrichtlinie)

Inhaltsverzeichnis

Präambel

Diese Programmierrichtlinie dient dem Ziel, die Interessen der Mitgliedsfirmen der FVA e. V., der Forschungsstellen und der FVA Software und Service GmbH so abzubilden, dass die Forschungsstellen in der Lage sind, die Programmieraufgaben mit wissenschaftlichem Bezug und einem in Forschungsprojekten darstellbaren Aufwand zu leisten und als Ergebnis Software für die Weiterentwicklung der FVA-Workbench ebenso wie für die Verwendung in Berechnungssystemen der Mitgliedsfirmen entsteht.

Eine Forschungssoftware unterliegt denselben Qualitätsbedingungen wie eine Anwendungssoftware. Dies bezieht sich vor allem auf die Aspekte Zuverlässigkeit, Fehlerfreiheit und das Vorhandensein einer ausreichenden Anzahl relevanter Testfälle.

Die Programmierrichtlinie enthält deshalb Qualitätskriterien und Anforderungen, denen die im Rahmen von FVA-Softwareprojekten entwickelte Software genügen muss, nicht zuletzt um eine langfristige Weiterentwicklung zum Nutzen der Anwender und FVA-Mitglieder zu ermöglichen.

0 Allgemeine Einführung

In der Regel bestehen Forschungsprojekte (mit dem Ziel, das Wissen in Form von Software anwendbar zu machen) aus zwei Themenblöcken. Zum einen aus dem theoretischen Forschungsanteil, in dem die Grundlagen erarbeitet sowie beschrieben werden und zum anderen aus dem Teil, in dem die Erkenntnisse in Form einer Software implementiert werden. Diese Anteile müssen beide bereits in der Antragsphase mit realistischen Zeitbedarfen berücksichtigt werden, sodass ein Gesamt-Zeitbedarf für Forschungsarbeit und Softwareentwicklung deutlich wird (s. Kapitel 1.1.3.3.).

Die im Folgenden beschriebenen Prozesse beziehen sich auf die Teile eines Forschungsprojektes, die sich mit der Umsetzung der Ergebnisse in eine Software befassen, d. h. die Erstellung Modularer Softwarebausteine (MSB) entsprechend der folgenden Abbildung. Diese sollen helfen, die Arbeiten im Detail zu planen und das Projekt erfolgsorientiert abzuschließen.

In der folgenden Abbildung 1 ist das Grundprinzip der Nutzung von in FVA-Forschungsvorhaben entwickelter Software sowohl in der FVA-Workbench (WB) als auch in den Berechnungsumgebungen der Mitgliedsfirmen der FVA e. V. dargestellt.

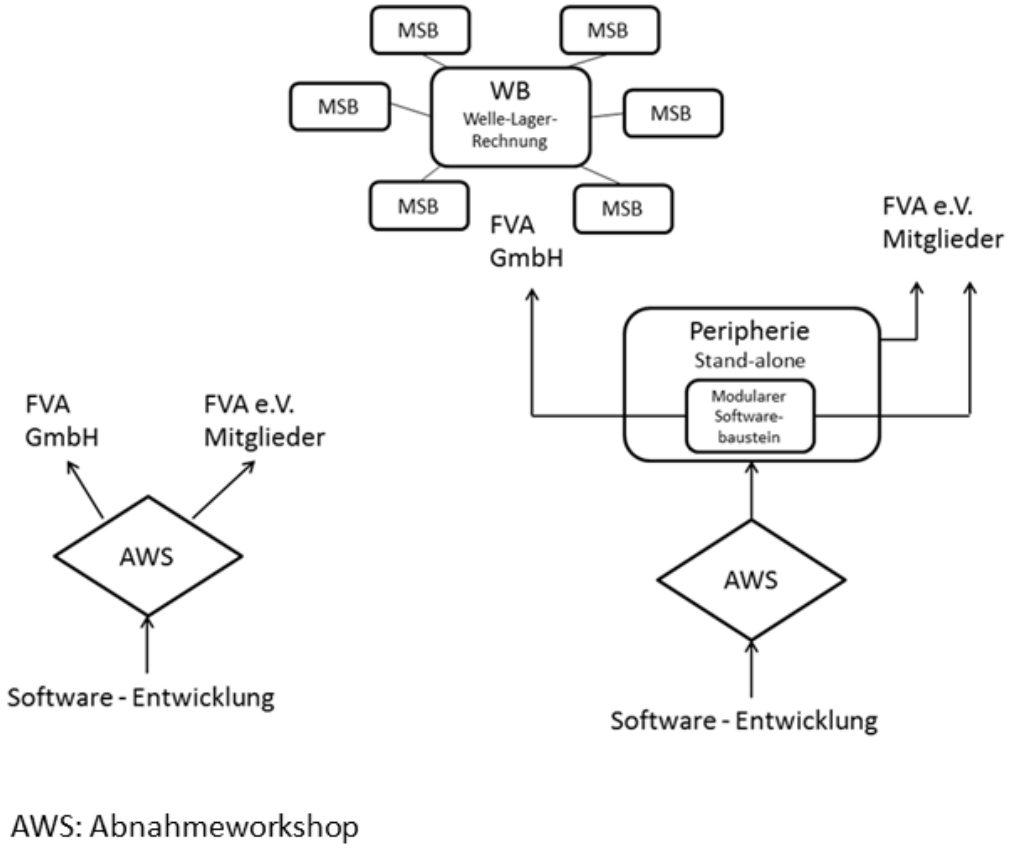


Abbildung 1: Nutzung von in FVA-Forschungsvorhaben entwickelter Software

Werden im Rahmen von Forschungsvorhaben Modulare Softwarebausteine erstellt, wird durch die Beachtung dieser Richtlinie zum einen sichergestellt, dass sie sich in die modular strukturierte FVA-Workbench im Rahmen eines Transferprojektes integrieren lassen, zum anderen stehen den FVA e. V. Mitgliedsfirmen sowohl die Modularen Softwarebausteine zur Ansteuerung aus einer eigenen

Softwareumgebung als auch die Modularen Softwarebausteine inklusive Peripherie als lauffähige Stand-alone-Version zur Verfügung.

Die Peripherie zu den Modularen Softwarebausteinen wird an den Forschungsstellen erstellt, um die modularen Softwarebausteine im Rahmen der im Folgenden beschriebenen, entwicklungsbegleitenden Testphasen auf methodische Richtigkeit prüfen zu können.

Jede Abweichung von dieser Richtlinie ist von der projektbegleitenden Arbeitsgruppe (AG) freizugeben und zu dokumentieren.

1 Ablauf der Softwareerstellung im Rahmen von FVA-Forschungsvorhaben

Die Erstellung von Software im Rahmen von FVA-Forschungsvorhaben folgt dem in Abbildung 2 dargestellten Ablauf. Wesentliche Grundelemente des Ablaufes sind:

- die exakte Definition der Anforderungen an die Software in Form einer weitgehend standardisierten Anforderungsliste (Kapitel 1.1.3.2.).
- die Sicherstellung einer mindestens hinreichenden Umsetzungsqualität durch einen Alpha-, Beta- und einen abschließenden Abnahmeworkshop (Kapitel 1.2.)
- die Bereitstellung der Software durch die Forschungsstellen in einem Versionsverwaltungssystem, z. B. GIT.

Im Versionsverwaltungssystem steht die Software den Mitgliedfirmen der FVA e. V. zur direkten Nutzung oder Einbindung in eigene Programmsysteme ebenso zur Verfügung wie der FVA Software & Service GmbH zur Integration in die FVA-Workbench.

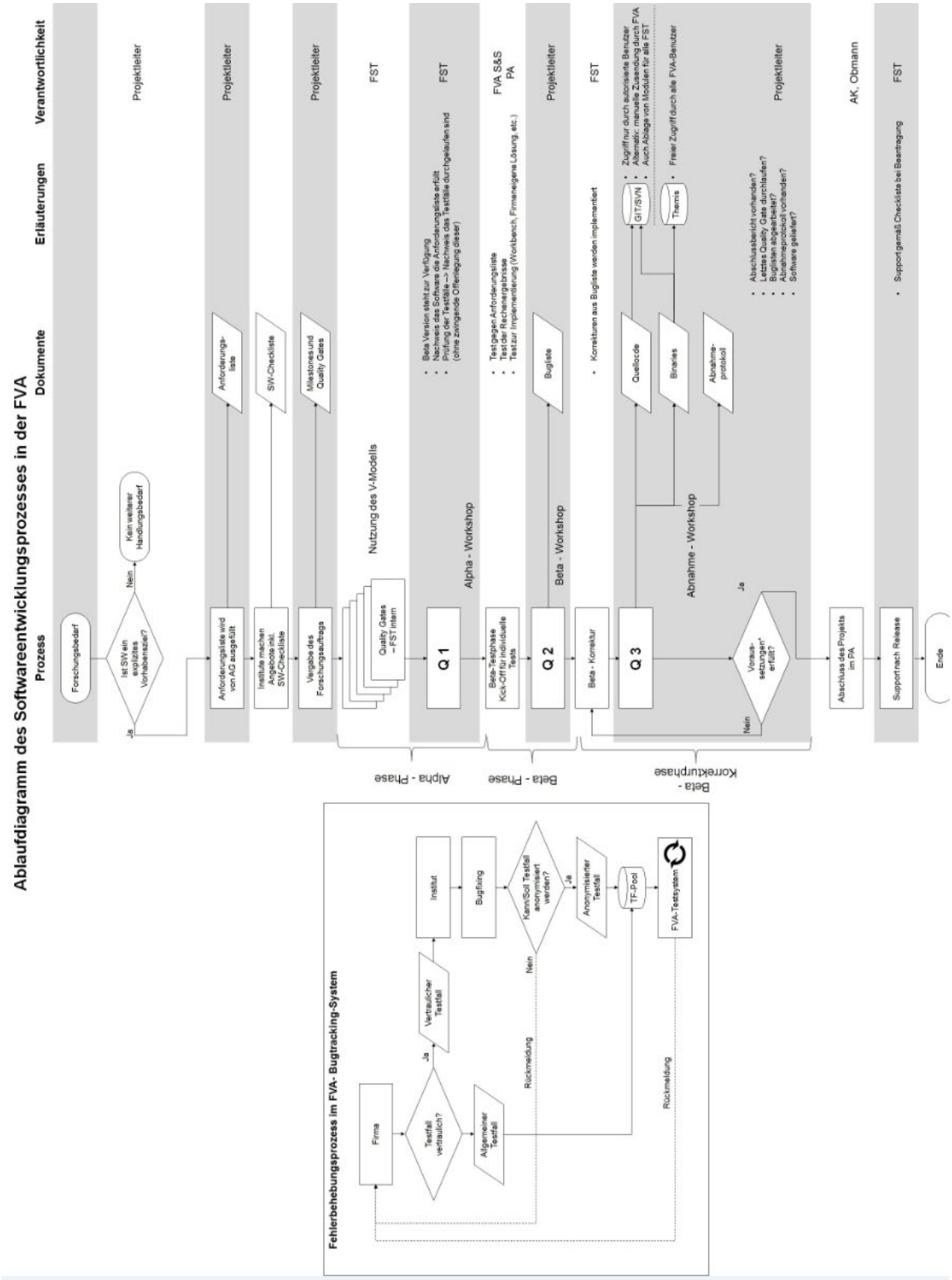


Abbildung 2: Ablauf der Softwareentwicklung im Rahmen von FVA Forschungsprojekten

1.1 Rahmenbedingungen der Antragsstellung

Folgende Rahmenbedingungen sind für die Beantragung, Durchführung und den Abschluss von Vorhaben, mit dem Ziel ein Forschungsergebnis in ein Berechnungsprogramm zu transformieren, zu beachten.

1.1.1 Grundregeln

1.) Während der Antragsphase zu einem Forschungsvorhaben mit dem Ziel der Softwareerstellung muss, in der Regel im Rahmen einer AG-Sitzung aus Industrievertretern, FVA Software & Service GmbH und Forschungsstellen, neben dem Forschungsantrag die Anforderungsliste an Software erarbeitet werden.

2.) Die projektbegleitende Arbeitsgruppe hat in Bezug auf die Programmerstellung in einem Forschungsvorhaben folgende Aufgaben:

   a. Definition der Anforderungen an das Programm (Anforderungsliste)
   b. Fachliches Beratungsgremium für den durchführenden Sachbearbeiter
   c. Kontrollgremium, das die Abarbeitung des Projektes überwacht und sicherstellt
   d. Qualitätssicherungsgremium, das sicherstellt, dass die im Rahmen des Projektes entwickelte Software den Anforderungen der Industrie genügt
   e. Die Abnahme der Software anhand der Anforderungsliste

3.) Bei Forschungsvorhaben, die die Erstellung eines Programms beinhalten, ist zusätzlich zum und gleichzeitig mit dem FVA-Forschungsantrag die Anforderungsliste vorzulegen.

4.) Bereits während der Alpha- und Beta-Workshops zur Überprüfung des Entwicklungsstandes des Programms ist die Funktionsfähigkeit des Rechenkerns durch genügend Rückmeldungen aus der industriellen Anwendung abzusichern (siehe auch Anlage [2]).

5.) Werden für die Parametrisierung Größen verwendet, die bereits im FVA-Workbench-Datenmodell zur Anwendung kommen, so ist die dort verwendete Nomenklatur anzuwenden. Es sollte unbedingt vermieden werden, dass gleiche physikalische Größen mit unterschiedlichen Bezeichnungen mehrfach implementiert werden.

6.) Neue Parameter oder Parameterbezeichnungen sind nur nach Information der FVA Software & Service GmbH einführbar. Damit wird die Aufnahme in das FVA-Workbench-Datenmodell möglich. Hierzu sind Benennung, Einheit, Formelzeichen und Kurzbeschreibung notwendig.

7.) Das Projekt zur Integration der Software in die FVA-Workbench stellt ein separates Projekt der FVA Software & Service GmbH dar. Es ist parallel zum Forschungsvorhaben zu beantragen. Eine enge Abstimmung im Zeitplan der Vorhaben ist anzustreben,

8.) Die Erstellung von grafischen Oberflächen wird von der FVA e. V. nur im Rahmen der FVA-Workbench unterstützt.

1.1.2 Verantwortlichkeiten bei Softwareintegration in die FVA-Workbench

Der Antragsteller, in der Regel die projektbegleitende Arbeitsgruppe, hat die Verantwortung für die Funktionsfähigkeit des Rechenkerns und die Schnittstellen zur Ein- und Ausgabe. Diese werden in der Anforderungsliste definiert. Wird eine spätere Integration in die FVA-Workbench als Anforderung spezifiziert, unterstützt und berät die Forschungsstelle die FVA Software & Service bei fachlichen Fragen des Tests der Integration. Hierfür werden sowohl Rechenkerntests (standardisiertes Testfallformat der FVA-Testbench unter Prüfung der langfristig unterstützten Schnittstellen für FVA-Rechenkerne) sowie FVA-Workbench-Beispiele (bei Integration in die FVA-Workbench: auch hier ist die FVA-Testbench zu verwenden.) der FVA Software & Service zur Verfügung gestellt. Die Testergebnisse müssen bei Abweichungen in den Ergebnissen zur Referenz von der Forschungsstelle ggf. fachlich bewertet werden (siehe Abbildung 3).

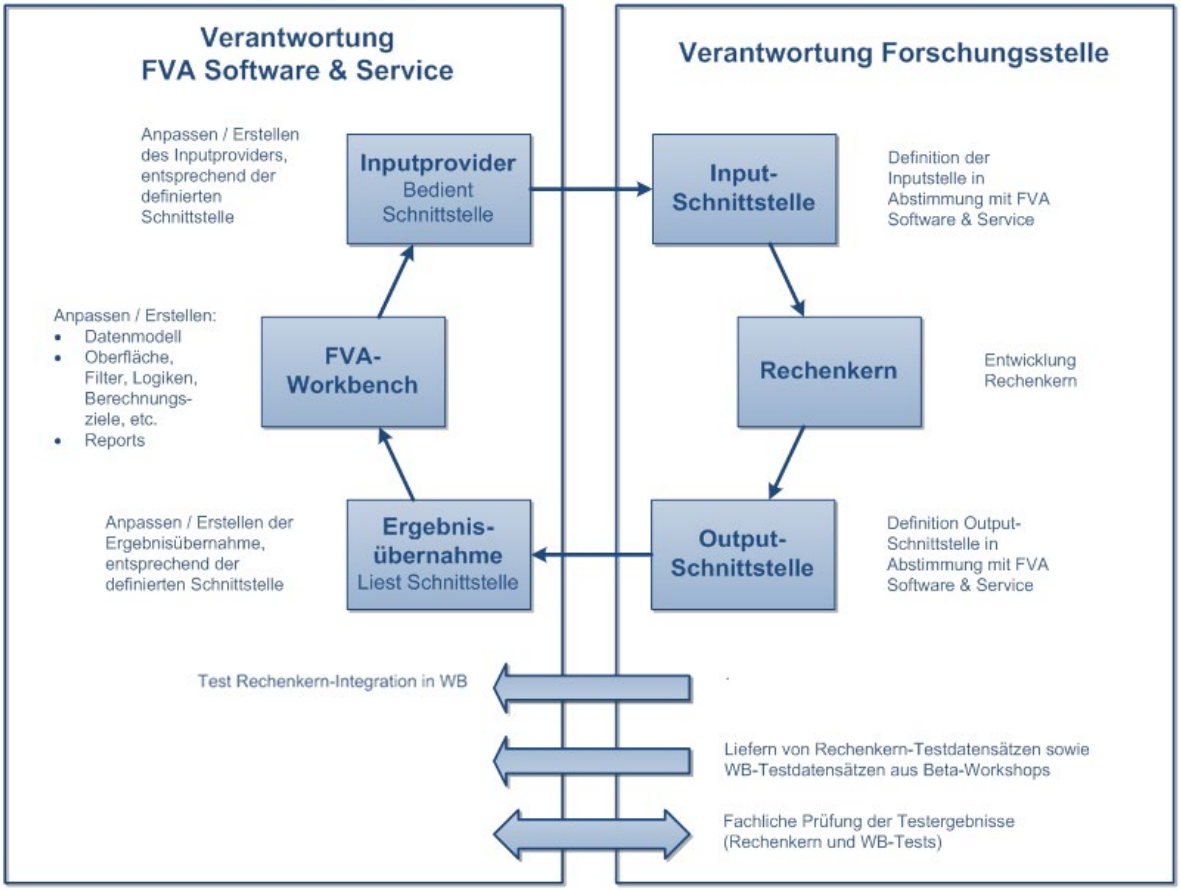


Abbildung 3: Verantwortlichkeiten bei Softwareintegration in die FVA-Workbench

1.1.3 Unterlagen

1.1.3.1 Forschungsantrag

Der Forschungsantrag enthält die Beschreibung der Forschungsarbeit an der Hochschule inkl. (Weiter-)Entwicklung einer FVA-Software unter Berücksichtigung dieser Richtlinie. Die der Software zugrunde liegenden Forschungsarbeiten sind im Rahmen eines Forschungsantrages zu beschreiben. Die Definition der Software soll hier auf Basis der vorangegangenen Forschungsarbeiten vorgenommen

werden. Hierbei sollen insbesondere die Berechnungsfunktionalitäten sowie die Schnittstellen zur Ein- und Ausgabe beschrieben werden.

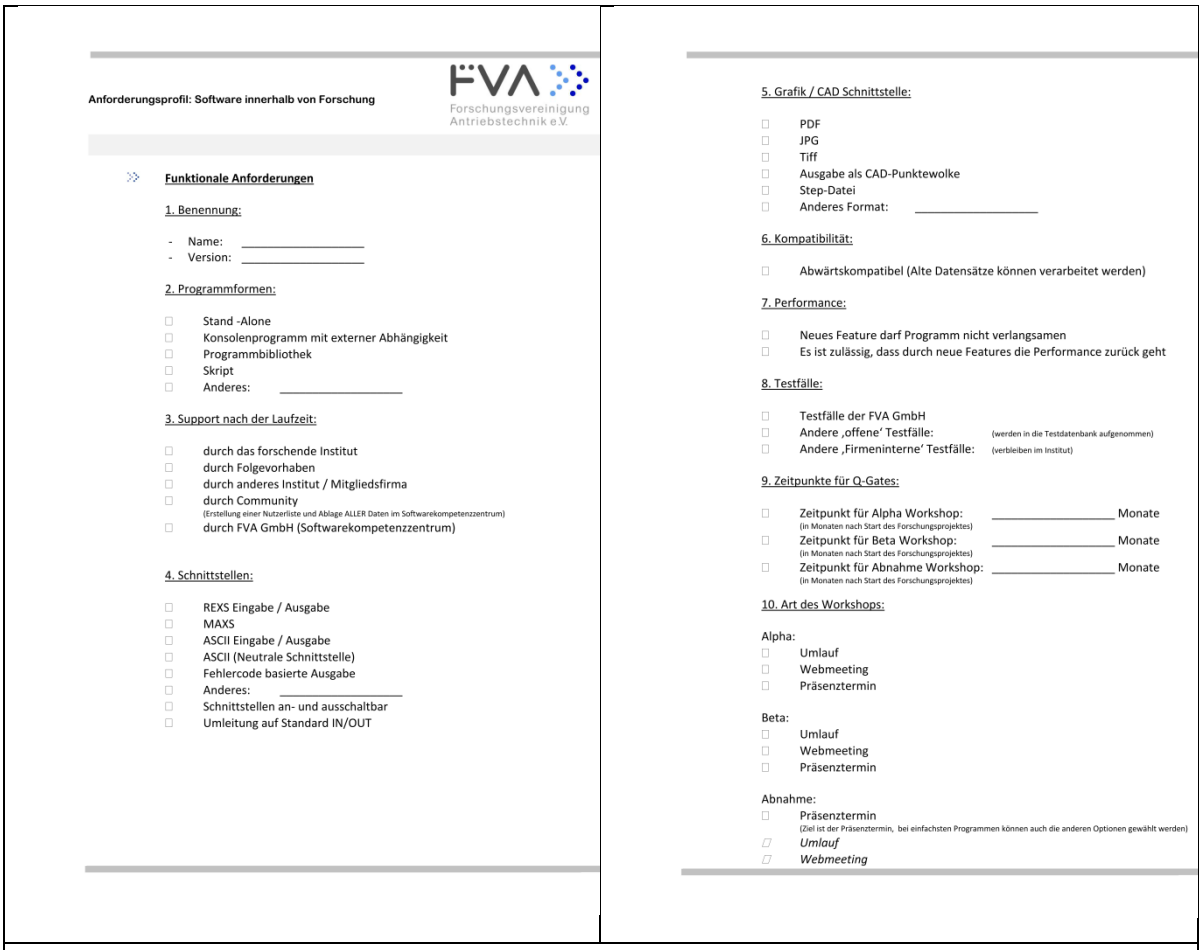
1.1.3.2 Anforderungsliste Software

Die Anforderungen an die Software, die im Rahmen von FVA-Forschungsvorhaben entwickelt wird, werden in einer Anforderungsliste definiert.

Damit ist eine Anforderungsdefinition in feinerer Abstufung als mit den bereits bekannten Entwicklungsstufen Methodenträger, Forschungssoftware und Anwendungssoftware möglich, die zur Orientierung im Anhang noch aufgeführt sind (s. Kapitel 3.2.).

Die Anforderungsliste Software ist Teil des Forschungsantrages. Die ersten beiden Seiten sind beispielhaft in Abbildung 4 dargestellt, die vollständige Anforderungsliste ist in Kapitel 3.1 zu finden. Sie wird in der Antragsphase erarbeitet und definiert unter anderem die folgenden Punkte:

- Entwicklungsstufe, die das Programm am Vorhabensende erreicht haben soll.
- Inhaltlich-funktionale Anforderungen an die Software.
- Ein- und Ausgabeformate.
- Festlegungen zu verwendeten Programmiersprachen.
- Integration in die FVA-Workbench gewünscht?
- GUI gewünscht/erforderlich?
- Ggf. Definitionen von Schnittstellen.
- …



Anforderungsprofil: Software innerhalb von Forschung

FVA Forschungsvereinigung Antriebstechnik e.V.

Funktionale Anforderungen

1. Benennung:

- Name: _____
- Version: _____

2. Programmformen:

☐ Stand -Alone
☐ Konsolenprogramm mit externer Abhängigkeit
☐ Programmbibliothek
☐ Skript
☐ Anderes: _____

3. Support nach der Laufzeit:

☐ durch das forschende Institut
☐ durch Folgevorhaben
☐ durch anderes Institut / Mitgliedsfirma
☐ durch Community
(Erstellung einer Nutzerliste und Ablage ALLER Daten im Softwarekompetenzzentrum)
☐ durch FVA GmbH (Softwarekompetenzzentrum)

4. Schnittstellen:

☐ REXS Eingabe / Ausgabe
☐ MAXS
☐ ASCII Eingabe / Ausgabe
☐ ASCII (Neutrale Schnittstelle)
☐ Fehlercode basierte Ausgabe
☐ Anderes: _____
☐ Schnittstellen an- und ausschaltbar
☐ Umleitung auf Standard IN/OUT

5. Grafik / CAD Schnittstelle:

☐ PDF
☐ JPG
☐ Tiff
☐ Ausgabe als CAD-Punktewolke
☐ Step-Datei
☐ Anderes Format: _____

6. Kompatibilität:

☐ Abwärtskompatibel (Alte Datensätze können verarbeitet werden)

7. Performance:

☐ Neues Feature darf Programm nicht verlangsamen
☐ Es ist zulässig, dass durch neue Features die Performance zurück geht

8. Testfälle:

☐ Testfälle der FVA GmbH
☐ Andere ‚offene' Testfälle: (werden in die Testdatenbank aufgenommen)
☐ Andere ‚Firmeninterne' Testfälle: (verbleiben im Institut)

9. Zeitpunkte für Q-Gates:

☐ Zeitpunkt für Alpha Workshop: _____ Monate
(in Monaten nach Start des Forschungsprojektes)
☐ Zeitpunkt für Beta Workshop: _____ Monate
(in Monaten nach Start des Forschungsprojektes)
☐ Zeitpunkt für Abnahme Workshop: _____ Monate
(in Monaten nach Start des Forschungsprojektes)

10. Art des Workshops:

Alpha:
☐ Umlauf
☐ Webmeeting
☐ Präsenztermin

Beta:
☐ Umlauf
☐ Webmeeting
☐ Präsenztermin

Abnahme:
☐ Präsenztermin
(Ziel ist der Präsenztermin, bei einfachsten Programmen können auch die anderen Optionen gewählt werden)
☐ *Umlauf*
☐ *Webmeeting*

Abbildung 4: Anforderungsliste Software, Seiten 1 und 2 als Auszug

1.1.3.3 Projekt- und Zeitplan

Der Projekt- und Zeitplan ist Teil des Forschungsantrags. Für die Entwicklung der Software ist ein realistischer Projekt- und Zeitplan zu erstellen, der im Falle einer gewünschten Integration in die FVA-Workbench auch mit der FVA Software & Service (Entwicklung der Oberfläche) abgestimmt ist. Dieser muss konkrete Zeitpunkte beinhalten, an denen ein Ergebnis der jeweiligen Parteien vorliegen muss. Die qualitätssichernden Alpha-, Beta- und Abnahmeworkshops entsprechend der Anforderungsliste müssen als Meilensteine enthalten sein (vgl. Kapitel 1.2.1.).

Für die Realisierung der Software hat es sich bewährt, die Projektphasen wie folgt abzuschätzen (siehe Abbildung 5):

- Konzeptphase 30% der Projektlaufzeit
- Entwicklungsphase 20% der Projektlaufzeit
- Test- und Abnahmephase 50% der Projektlaufzeit

Bei diesen Werten handelt es sich um Richtwerte, die bei der Programmentwicklung gängige Praxis sind.

Konzeptphase, Erstellung von Spezifikationen:

Zu Beginn des Forschungsvorhabens soll die Softwarearchitektur bzw. die Eingliederung der neuen Programmteile in eine bestehende Software (bei Erweiterungen) entworfen und vorgestellt werden.

Entwicklungsphase, Implementierung:

Es soll ausreichend Zeit für die Entwicklung der Algorithmen sowie die Umsetzung in Quelltext eingeplant werden. Zu beachten ist vor allem der Zeitaufwand zur Dokumentation des Codes sowie zur Umsetzung einer umfassenden Fehlerbehandlung. Vielfach werden Anforderungen in den begleitenden Arbeitsgruppen- und Arbeitskreissitzzungen, auch zur Laufzeit des Vorhabens, noch geschärft.

Test- und Abnahmephase:

Die Abnahme erfolgt in Zusammenarbeit mit der projektbegleitenden Arbeitsgruppe, der das Programm im Rahmen eines strukturierten Workshops vorgestellt wird und die anschließend über einen definierten Testzeitraum das Programm anwenden und prüfen soll. Ergebnis des Abnahmeworkshops ist das Abnahmeprotokoll, welches alle Nachbesserungswünsche und -forderungen enthält und eine Freigabe des Programms, ggf. unter Auflagen, dokumentiert. Weitere Details hierzu sind im folgenden Kapitel 1.2 detailliert beschrieben.

1.1.3.4 Spezifikation für die Integration in die FVA-Workbench

Im Falle einer gewünschten Einbindung erstellt die FVA Software & Service GmbH einen eigenen Antrag zur Integration des Rechenkerns in die FVA-Workbench (Transferprojekt). Hierbei wird der Leistungsteil beschrieben und definiert. Notwendige Anpassungen im Datenmodell, der Oberflächen oder neue Berechnungswege werden hier beschrieben. Die Forschungsstelle steht für eine enge Abstimmung für die Projektdefinition zur Verfügung.

1.2 FVA-Entwicklungsprozess bei Forschung mit Softwareentwicklung

Um ein zielgerichtetes Vorgehen bei der Softwareentwicklung zu erreichen, ist es notwendig, den Entwicklungsprozess durch klar definierte Phasen zu strukturieren. Hierzu hat sich der V-Prozess in der Softwareentwicklung als Standardvorgehensweise etabliert.

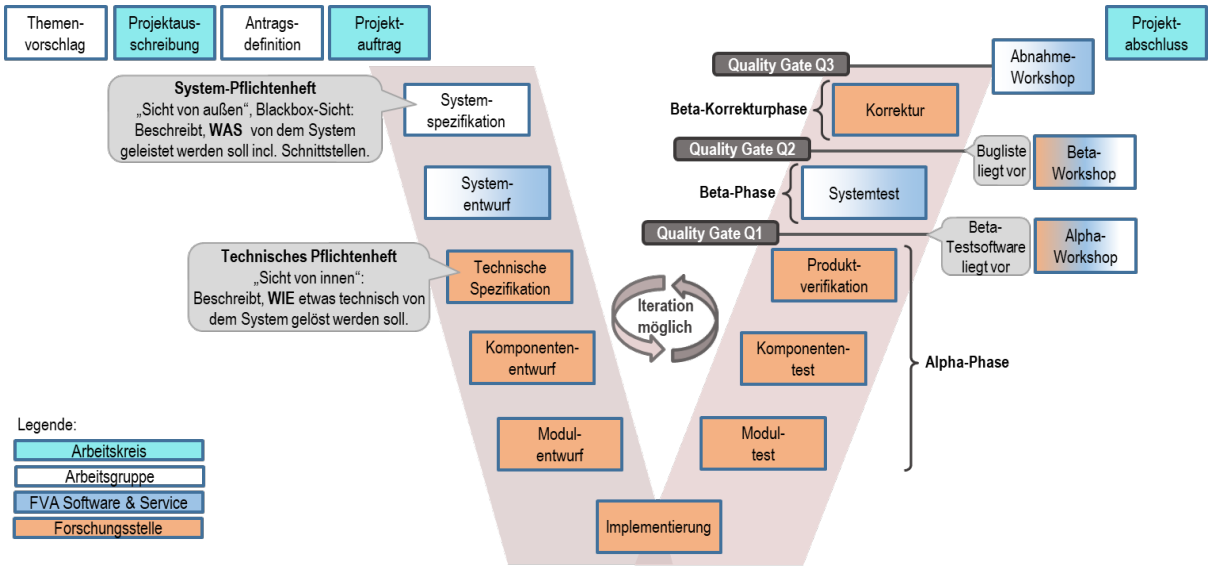


Abbildung 5: Vorgehen bei der Realisierung von FVA-Vorhaben mit Anteilen zur Programmerstellung oder -erweiterung

Die Beschreibung der Prozessschritte ist in Abbildung 6 dargestellt. Die Schritte ***Themenvorschlag*** bis zum ***Projektauftrag*** sind für alle FVA-Vorhaben analog abzuhandeln. Hierbei gibt es keinen Unterschied zwischen Vorhaben mit und ohne Softwareentwicklung. Detaillierte Informationen hierzu sind zu finden in Anlage [5].

1.2.1 Quality Gates

Quality Gates (QG) sind bestimmte Kontrollpunkte im Projekt, zu denen ein Ergebnis vorliegen muss bzw. das weitere Vorgehen auf Basis der Ergebnisse abgestimmt werden kann (s. auch Abbildung 6: Quality Gates). Die Zeitpunkte für die Quality Gates werden in der Anforderungsliste definiert.

- Quality Gate 1: Funktionsfähige Alpha-Version liegt vor, ist forschungsstellenintern getestet und geht damit in den Status einer Beta-Testsoftware über, welche von den Anwendern (Arbeitsgruppenmitgliedern und FVA Software und Service) in der anschließenden Beta-Phase gegen die Anforderungsliste geprüft werden kann.
- Quality Gate 2: Funktionsfähige Beta-Version liegt vor, die von den Anwendern gegen die Spezifikation geprüft wurde. Eine Bugliste wird vorgestellt.
- Quality Gate 3 entspricht dem Abnahmeworkshop: Programm mit behobenen Mängeln und Anmerkungen aus der Beta-Korrekturphase liegt vor. Die Bugliste ist abgearbeitet.

Die in Abbildung 5 dargestellte Vorgehensweise bietet ab dem Punkt „Systemspezifikation“ immer die Möglichkeit, Funktionen und Anforderungen iterativ anzupassen und zu detaillieren.

Diese Iterationen müssen personell und inhaltlich abgestimmt sein, sodass die Projektlaufzeit nicht ungeplant in die Länge gezogen wird. Zusätzlich gewünschte, im ursprünglichen Angebot aber nicht spezifizierte Funktionserweiterungen sollen in ein Folgeprojekt verschoben werden. Damit wird sichergestellt, dass diese Projekte ein definiertes Ende haben und allen am Projekt Beteiligten der jeweilige Status der Entwicklung transparent ist.

Abbildung 6: Quality Gates

1.2.2 Systemspezifikation

Die Systemspezifikation erläutert auf Grundlage des Forschungsantrages und der Anforderungsliste die gewünschten Funktionen der Software (WAS soll die Software können?) und beschreibt damit das geplante Ergebnis der Softwareentwicklung konkret.

Es wird empfohlen, bei der Systemspezifikation mit User-Stories zu arbeiten (siehe Anlage [3]).

1.2.3 Technische Spezifikation:

Hier wird auf der Grundlage der Anforderungsliste zur Softwareentwicklung aus der Antragsphase (vgl. Kapitel 1.1.3.2) festgelegt, WIE die Funktionen aus der Systemspezifikation umgesetzt werden. Dies beinhaltet u. a.:

- Schnittstellen,
- Parameter,
- System,
- Berechnungsgrenzen.

Dies erfolgt im Rahmen von Workflowbeschreibungen sowie der Beschreibung der Schnittstellen. In dieser Phase wird die Systemstruktur bestimmt und damit festgelegt, wie die Komponenten und Module in der Gesamtanwendung zusammenspielen.

1.2.4 Komponenten- und Modulentwurf (strukturierter Entwurf)

Der Komponenten- und Modulentwurf besteht aus Bausteinen, den Komponenten, die die Umsetzung der Funktionen aus der technischen Spezifikation beinhalten. Komponenten bestehen aus einzelnen Modulen, die sinnvolle funktionale Softwarebausteine darstellen.

Es werden jetzt Kriterien oder Testfälle definiert, gegen welche die Komponenten geprüft werden.

1.2.5 Qualitätssicherung / Testing

Um die Qualität der erstellten Software sicherzustellen und somit die langfristige Weiterentwicklung zu ermöglichen ist eine ausreichende Anzahl relevanter Testfälle zu erstellen. Art und Umfang sind mit der entsprechenden projektbegleitenden Arbeitsgruppe abzustimmen.

Mit der FVA-Testbench stellt die FVA ein Tool zur Verfügung, das die Erstellung, Verwaltung und automatisierte Ausführung von Testfällen sehr einfach gestaltet, sodass sich der Ersteller der Testfälle auf deren Inhalt konzentrieren kann. Die FVA-Testbench kann vor Erstellung der Testfälle von der FVA bezogen werden. Die Nutzung der FVA-Testbench wird empfohlen.

Das Testing findet dabei auf zwei Ebenen statt:

1. Rechenkernebene (methodische Richtigkeit)
   Während der Weiterentwicklung eines Rechenkerns sind kontinuierliche Tests wichtig, um die Qualität des Ergebnisses sicherzustellen.
   Ein Testfall auf Rechenkernebene besteht aus den benötigten Eingabedateien und den hinterlegten Referenzdateien. Es müssen alle Funktionen des Rechenkerns getestet werden, vgl. Kapitel 1.2.1.
2. Ebene FVA-Workbench
   Ein Testfall besteht hier aus einem Modell der FVA-Workbench (*.wbps-Datei) sowie den hinterlegten Referenzdateien. Diese Tests sind Bestandteil des Transferprojektes zur Integration des modularen Softwarebausteins in die FVA Workbench.

Alle Testfälle sind zu dokumentieren. Bevor ein Rechenkern freigegeben werden kann, sind alle Testfälle, inklusive aller notwendigen Dateien sowie die Ergebnisreports des Testsystems, an das FVA Software Kompetenzzentrum zu übergeben.

Das FVA-Software-Kompetenzzentrum prüft abschließend alle Testfälle. Erst nach Freigabe des Rechenkerns durch das FVA-Software-Kompetenzzentrum kann das Projekt erfolgreich abgeschlossen werden.

1.2.6 Modul-, Komponenten- und Systemtest (Alpha-Testphase)

Nachdem die Module entwickelt wurden, sind diese den Modultests zu unterziehen. Jedes Modul sollte für sich getestet werden, um Fehler oder Fehlfunktionen auszuschließen. In dieser Phase ist auch die Fehlerbehandlung, welche später im Gesamtsystem benötigt wird, fertigzustellen und im Rahmen des Modultests zu validieren.

Nun ist zu prüfen, ob die einzelnen Module im Verbund als Komponente ein kontrolliertes Verhalten zeigen. Die in der Komponentenentwurfsphase definierten Testfälle sollen jetzt geprüft werden und bei Bedarf der Komponentenentwurf angepasst werden.

Neben der Einzelfunktion der Komponenten und Module muss auch das Zusammenwirken der Komponenten überprüft werden, sodass die in der Systemspezifikation definierten Anforderungen erfüllt sind (Systemtest). Diese Tests entsprechen der Alpha-Testphase, die forschungsstellenintern durchgeführt wird. Nach dem abschließenden Alpha-Workshop (Quality Gate 1), in dem die Forschungsstellen die Software und die Berechnungsfälle der Arbeitsgruppe vorstellen, kann die Beta-Testphase beginnen.

1.2.7 Systemtest auf Anwenderebene (Beta-Testphase)

In der Beta-Testphase wird das System durch Anwender (z. B. Mitglieder der Arbeitsgruppe) auf korrekte Funktion und Übereinstimmung mit der Anforderungsliste geprüft. Auch hier soll auf die in der Systementwurfsphase definierten Testfälle zurückgegriffen werden. Festgestellte Fehler werden von den Testern spätestens zwei Wochen vor dem Beta-Workshop-Termin an die Forschungsstellen gemeldet und dort gesammelt. Im Beta-Workshop (Quality Gate 2) wird von der Forschungsstelle eine kommentierte Bugliste vorgestellt, die in der verbleibenden Zeit (Beta-Korrekturphase) bis zum Abnahmeworkshop von den Forschungsstellen abgearbeitet werden muss.

1.2.8 Abnahme-Workshop und Projektabschluss:

Vor dem Abnahme-Workshop wird von den Instituten die Bugliste abgearbeitet.

Im Abnahme-Workshop werden folgende Punkte überprüft:

- Erfüllung der Anforderungsliste
- Abarbeitung der Bugliste
- Erfüllung der Ziele des Forschungsantrages
- Bereitstellung der Software im Versionsverwaltungssystem
- Vorhandensein der Dokumentation, u. a. der Entwurf des Forschungsheftes, mit Berechnungsbeispielen
- Vorhandensein von Testfällen
- Datenblatt Rechenkern-Release [1]

Die Workshop-Teilnehmer müssen während des Workshops einstimmig beschließen, dass die o. g. Punkte erfüllt sind. Damit ist das Projekt aus Sicht der projektbegleitenden Arbeitsgruppe abgeschlossen.

Die erfolgreiche Abnahme ist dem FVA-Software-Kompetenzzentrum in Form eines Abnahmeprotokolls mitzuteilen. Dieses muss vom Projektleiter der projektbegleitenden Arbeitsgruppe gegengezeichnet sein (Vorlage siehe Anlage [2]).

1.3 Pflege und Anwenderunterstützung zum Programm durch die Forschungsstellen

Während der Laufzeit des zugehörigen Forschungsvorhabens steht mit dem Sachbearbeiter an der Forschungsstelle ein Ansprechpartner für die Firmenvertreter zur Verfügung, der zu Fragen zum Programm Auskunft geben kann.

Für die Zeit nach Abschluss des Forschungsvorhabens ist die Programmbetreuung zusätzlich zu planen. Die Berichtspflicht endet für die Forschungsstelle mit dem Ende des Forschungsvorhabens. Um Anwenderfragen aus der Industrie auch nach dem Ende des Vorhabens aufnehmen zu können, sollen rechtzeitig geeignete Maßnahmen aus der folgenden Auswahl benannt werden:

1) Das Programm wird vom FVA-Software-Kompetenzzentrum vollständig inkl. der Quellen und sämtlicher Dateien, die zur Erstellung des ausführbaren Programms notwendig sind, archiviert. Neben den Quelldateien soll auch eine vollständige ausführbare Version abgelegt werden, die die Programmdokumentation bzw. das Forschungsheft, die Unterlagen des Abnahmeworkshops und die verwendeten Beispieldateien enthält.

2) Nach Ende des Vorhabens soll das Forschungsinstitut für Anwenderfragen in einem zumutbaren Rahmen zur Verfügung stehen. Die Möglichkeit zur Anwenderunterstützung wird begünstigt, wenn der Sachbearbeiter noch am Institut tätig ist. Ebenso positiv ist die Nennung eines festen Ansprechpartners am Institut zu werten, der auch über längere Zeit dort tätig ist. Der Ansprechpartner am Forschungsinstitut ist im Abnahmeworkshop zu benennen.

3) Weiterer Forschungsbedarf muss frühzeitig von den Mitgliedsfirmen der FVA e. V. im Rahmen von Gremiensitzungen festgestellt und daraus ein Folgevorhaben beantragt werden. Damit können Programmwartung und -support bei entsprechenden Anwenderwünschen durch den Sachbearbeiter forschungsstellenseitig eingeplant und geleistet werden.

4) Wird das Vorhaben nicht fortgesetzt bzw. besteht am Forschungsinstitut nicht die Möglichkeit der weiteren Anwenderunterstützung, kann grundsätzlich ein anderes Forschungsinstitut den Support zu übernehmen oder die FVA-Community für Fragen zur Verfügung stehen.

Für die Kommunikation und Nachverfolgung von Anwenderfragen und Fehlerhinweisen nach der Freigabe durch das FVA-Software-Kompetenzzentrum wird das Bugtracking-System (BTS) verwendet. Hier wird durch das FVA-Software-Kompetenzzentrum ein Zugang für die Forschungsstelle (zweckmäßig für den Sachbearbeiter) erstellt. Eine Einweisung in das Programm und in das Vorgehen zur Bearbeitung von BTS-Einträgen wird entsprechend durch die FVA-Software-Kompetenzzentrum durchgeführt.

Um eine zentrale Fehler- und Anfragenerfassung zu ermöglichen, müssen auch Anfragen, die unter Umständen erst von den Forschungsstellen beantwortet werden können, daher zunächst an den Support (E-Mail: support@fva-service.de, Tel.: +49 69 6603 1991) gerichtet werden.

1.4 Beteiligte Organisationen bei der FVA-Softwareentwicklung und deren Rollen

Damit das Ergebnis Ihrer Forschungsarbeit, das von allen Mitgliedsfirmen der FVA finanziert ist, auch allen Mitgliedsfirmen zur Verfügung steht, hat die FVA weitere Organisationen eingebunden, die Sie bei der Entwicklung der Software und bei der Nachbereitung (Einführung der Software) in den

Mitgliederkreis unterstützen. In diesem Kapitel sind diese benannt und kurz beschrieben. Für weitere Informationen steht Ihnen die FVA-Geschäftsstelle zur Verfügung.

1.4.1 FVA e. V.

Rolle: Koordination übergeordnetes Projektcontrolling

Strukturierung und Koordination der FVA-Softwareentwicklung

- Beratung und Betreuung der FVA-Gremien
- Betreuung und Ausbau der FVA-Softwareinfrastruktur
- Kommunikation und Öffentlichkeitsarbeit zur FVA-Software
- Überwachung und Regelung der rechtlichen Rahmenbedingung für FVA-Software
- Beratung des wissenschaftlichen Beirates und des Vorstandes der FVA
- Ausrichtung der FVA-Softwareentwicklung, sodass die Strategie der FVA unterstützt wird.

1.4.2 FVA-Software-Kompetenzzentrum

Rolle: Beratungs-, Qualitätssicherungs- und Abnahmeinstanz

1) Unterstützung von Anwendern in der Industrie bei Fragen zu FVA-Programmen
   - Installation und Anwendung auch von Altprogrammen
   - Inhaltliche Fragen zu theoretischen Grundlagen und der Anwendung auf Praxisfragestellungen
   - Fragen zur Bedienung (bisherige Benutzeroberflächen und aktuelle FVA-Workbench)
   - Informationen zum Quelltext und zur Übersetzung/Portierung

2) Organisation der FVA-Softwareentwicklung
   - Pflege und Anpassung dieser Richtlinie in Abstimmung mit der AG SuP
   - Formale Abnahme von Programmen (Vollständigkeit, Dokumentation, Einhaltung der Vorgaben gemäß „FVA-Richtlinie für die Beantragung und Durchführung von Softwareentwicklungsprojekten“)
   - Verwaltung des FVA-Software-Versionsverwaltungssystems in Themis
   - Strukturierung der Wartungs- und Supportinfrastruktur
   - Aktualisierung und Pflege der Software in Themis
   - Bereitstellung der FVA-Programme für die Mitgliedsfirmen im FVA-Software-Versionsverwaltungssystem

3) Beratung anderer Institute bei der Erstellung von FVA-Programmen
   - Unterstützung bei Problemen in der Entwicklung
   - Beratung zur Spezifikation erforderlicher FVA-Workbench-Erweiterungen

1.4.3 FVA Software und Service GmbH

Rolle: Weiterentwicklung der FVA-Workbench

- Zuständig für FVA-Workbench-Entwicklung
- Pflege des übergeordneten FVA-Workbench-Datenmodells

1.4.4 Arbeitsgruppen

Rolle: Kontroll-, Projektpartner- und Steuergremium

1) Definition des Forschungs- und/oder Erweiterungsbedarfs
2) Festlegung und Überwachung der Zeit- bzw. Projektpläne und Ressourcenmanagement
3) Begleitung der Forschungsstelle bei der Durchführung ihrer Softwareentwicklung (Forschungsvorhaben)
4) Anforderungsdefinition (Anforderungsliste Software, vgl. Kapitel 1.1.3.2)
5) Abnahme der Ergebnisse
6) Verantwortung für die Abnahme der Leistungen der Forschungsstelle
7) Sicherstellen der technischen Richtigkeit und Zuverlässigkeit der entwickelten Berechnungsansätze.

1.4.5 Arbeitskreise

Rolle: Auftraggeber und Abnahmeinstanz

1) Beratungsgremium für die Arbeitsgruppen
2) Formale Institution zur Beantragung bzw. zur Abnahme der Forschungsergebnisse
3) Informationsgremium für FVA-Mitglieder.

2 Allgemeine Pflichten und Anforderungen an die Softwareentwicklung

Bei der Codeerstellung sind die „Grundsätze guter Softwareentwicklung“ einzuhalten, wie sie beispielsweise in Clean Code-Prinzipien dargelegt sind [4].

Das FVA-Software-Kompetenzzentrum steht in allen Phasen der Softwareentwicklung beratend zur Verfügung.

2.1 Disclaimer

Bei Ausführung eines Programms auf der Konsole und auf dem Berechnungsausdruck ist ein Disclaimer darzustellen, um die Haftungsbeschränkung des FVA e. V. sicherzustellen. Der Disclaimer muss vom Nutzer quittiert werden und für Serienberechnungen in der config-Datei abschaltbar sein.

Folgender Text ist zu verwenden:

> BITTE LESEN SIE DIESEN DISCLAIMER SORGFÄLTIG DURCH, BEVOR SIE DIESES FORSCHUNGSERGEBNIS (DIGITALER METHODENTRÄGER) LADEN ODER IN BETRIEB NEHMEN. INDEM SIE DEN DIGITALEN METHODENTRÄGER VERWENDEN, ERKLÄREN SIE IHR EINVERSTÄNDNIS MIT DEN BESTIMMUNGEN DER NACHSTEHENDEN NUTZTUNGSREGELUNG.
>
> **Anwendungsbereich**
>
> Der vorliegende DIGITALE METHODENTRÄGER ist das Ergebnis eines Forschungsprojektes, welches im Rahmen der Industriellen Gemeinschaftsforschung durch die Forschungsvereinigung Antriebstechnik e.V. (FVA) entwickelt wurde. Der DIGITALE METHODENTRÄGER dient lediglich dazu, im Rahmen der Projektarbeit Theorien, Ergebnisse und Algorithmen zu überprüfen und zu demonstrieren bzw. zu validieren. Sie können den DIGITALEN METHODENTRÄGER herunterladen, kopieren und im Rahmen der industriellen Gemeinschaftsforschung ausführen. Die FVA übernimmt keine Gewähr für die Ausführbarkeit, Anwendbarkeit oder für sonstige Eigenschaften. Es erfolgt auch keine Prüfung durch die FVA, ob der DIGITALE METHODENTRÄGER frei von Rechten Dritter ist. Die Anwendung des DIGITALEN METHODENTRÄGERS erfolgt ausschließlich in Eigenverantwortung des Nutzers.
>
> **Gewährleistungsausschluss**
>
> Es besteht keinerlei Gewährleistung für den DIGITALEN METHODENTRÄGER, soweit dies gesetzlich zulässig ist. Sofern nicht anderweitig schriftlich bestätigt, stellen die Urheberrechtsinhaber und/oder Dritte das Programm so zur Verfügung, „wie es ist“, ohne irgendeine Gewährleistung, weder ausdrücklich noch implizit, einschließlich – aber nicht begrenzt auf – die implizite Gewährleistung der Marktreife oder der Verwendbarkeit für einen bestimmten Zweck. Das volle Risiko bezüglich Qualität und Leistungsfähigkeit des Programms liegt bei Ihnen. Sollte sich das Programm als fehlerhaft herausstellen, liegen die Kosten für notwendigen Service, Reparatur oder Korrektur bei Ihnen.

Haftungsbegrenzung

In keinem Fall, außer wenn durch geltendes Recht gefordert oder schriftlich zugesichert, ist irgendein Urheberrechtsinhaber oder irgendein Dritter, der den DIGITALEN METHODENTRÄGER übertragen hat, Ihnen gegenüber für irgendwelche Schäden haftbar, einschließlich jeglicher allgemeiner oder spezieller Schäden, Schäden durch Seiteneffekte (Nebenwirkungen) oder Folgeschäden, die aus der Benutzung des Programms oder der Unbenutzbarkeit des Programms folgen (einschließlich – aber nicht beschränkt auf – Datenverluste, fehlerhafte Verarbeitung von Daten, Verluste, die von Ihnen oder anderen getragen werden müssen, oder dem Unvermögen des Programms, mit irgendeinem anderen Programm zusammenzuarbeiten), selbst wenn ein Urheberrechtsinhaber oder Dritter über die Möglichkeit solcher Schäden unterrichtet worden war.

Weitergabe

Weiterverbreitete Exemplare müssen das Copyright „Copyright Forschungsvereinigung Antriebstechnik e.V.“ sowie den Gewährleistungsausschluss/Haftungsausschluss in der Dokumentation und/oder anderen Materialien, die mit dem Exemplar verbreitet werden, enthalten.

Alle Werbematerialien, die Eigenschaften oder die Benutzung erwähnen, müssen die folgende Bemerkung enthalten:

„Dieses Produkt enthält einen DIGITALEN METHODENTRÄGER, der im Auftrag der Forschungsvereinigung Antriebstechnik e.V. und Ihren Mitgliedern entwickelt wurde.“

2.2 Multi-User-Fähigkeit

Programme, die entsprechend dieser Richtlinie neu erstellt werden, sollen Multi-User-fähig sein. Unter einem Multi-User-fähigen Programm ist in diesem Zusammenhang ein Programm zu verstehen, von dem mehrere Instanzen gleichzeitig auf einem System laufen können, ohne dass sich dabei die Daten der verschiedenen Instanzen gegenseitig überschreiben oder vermischen.

2.3 Plattformunabhängigkeit

Programme, die auf Grundlage dieser Richtlinie in portierbaren Sprachen entsprechend der Anforderungsliste neu erstellt werden, sollen auf Windows und Linux-Systemen lauffähig sein.

2.4 Programmerstellung und -architektur

Vor der Programmierung der Funktionalität des Berechnungskerns ist der Programmaufbau zu entwerfen und in Diagrammen als Ablaufplan bzw. als Programmstruktur darzustellen. Bei Erweiterungen vorhandener Programme ist die Einordnung der neuen Funktionalität in die bestehende Struktur zu dokumentieren.

Grundsätzlich ist die Gliederung des Programmes abhängig vom Funktionsumfang. Das Programm ist in Module zu gliedern, die größere Berechnungsaufgaben erfüllen (vgl. Kapitel 1.2.4.). Jedes Modul muss über eine klare Schnittstelle verfügen, über die sämtliche Eingangs- und Ergebnisdaten übergeben werden. Ein Datentransfer „im Hintergrund“ (z. B. über Common-Blöcke) ist nicht zulässig.

Jedes Modul ist in weitere Unterprogramme und Routinen zu ordnen. Innerhalb von Modulen ist der Datenaustausch zwischen den Routinen auch über globale Variablen zulässig, soweit die Notwendigkeit dafür begründet werden kann.

Ein Programmmodul zeichnet sich aus durch:

- eine klare explizite Schnittstelle und Formalparameterliste der Berechnungsroutine
- Ein-/Ausgabe (Daten-)Schnittstelle (Einlesen der Eingabedatei, Schreiben der Ausgabedatei)
- Recheninhalt in Form von Subroutine/Objekt/Module

Zu jedem Berechnungsmodul sind Routinen zum Einlesen der Eingabedaten sowie zur Ausgabe der Berechnungsergebnisse in Textform und als Grafik zu erstellen wie in der Anforderungsliste definiert.

Falls notwendig kann aus Berechnungs-, Einlese- und Ausgaberoutinen durch einen zusätzlich zu implementierenden Steueralgorithmus ein ausführbares Programm erstellt werden.
Zwei Formen der ausführbaren Programmmodule sind gängig: Ausführbare Dateien und dynamische Programmbibliotheken.

Ausführbare Datei, z. B. .exe-Datei

Bei der Integration einer ausführbaren Datei in ein Gesamtsystem benutzt das Programmmodul normalerweise eine Dateischnittstelle. Wenn dieses Modul aber mit mehreren anderen Modulen Daten gleichzeitig austauschen oder von mehreren Modulen benutzt werden soll, führt eine Aufteilung eines Gesamtprogrammes in ausführbare Dateien zu Problemen.

Dynamische Programmbibliothek (z. B. .dll-Datei)

Die als dynamische Bibliothek kompilierten Programmmodule bieten den Vorteil einer flexiblen Nutzung. Die dynamische Bibliothek wird nur zur Laufzeit gebunden und nur eine Kopie wird in den Hauptspeicher geladen. Unterschiedliche Prozesse benutzen die einzige Kopie für eigene Funktionen. Ist die Schnittstelle der dynamischen Bibliothek gut definiert, so hat ein Austausch der Bibliothek z. B. bei einem Update, keinen Einfluss auf das Gesamtsystem. Die Aufteilung in mehrere dynamische Bibliotheken ist für ein Projekt sehr günstig, weil der Rechenkern in unabhängigen Funktionalitäten zerlegt werden kann.

2.5 Programmiersprache

Die Programmiersprache wird in der Anforderungsliste definiert.

2.6 Bibliotheken

Zur Verarbeitung der Ein- und Ausgabedaten sind im gesamten Programm einheitliche, standardisierte Bibliotheksfunktionen zu verwenden. Die verwendete Ausgabebibliothek muss die Textausgabe auf Basis von Maskendateien/Templates erstellen. Damit ist sicherzustellen, dass sämtliche Ausgabetexte unabhängig vom Programmcode (ohne erneutes Kompilieren und Linken) geändert werden können. Alle verwendeten Bibliotheken müssen frei von Rechten Dritter sein. Erlaubt sind alle GPL- und LGPL-lizensierten Bibliotheken. Die Quellen der verwendeten Bibliotheken müssen genannt werden.

2.7 Programmaufbau und -schnittstellen

Die zu verwendenden Schnittstellen werden in der Anforderungsliste definiert.

Die Berechnungsprogramme können vom Anwender über die Dateischnittstellen bedient werden und über diverse Schnittstellen mit der FVA-Workbench oder firmeneigenen Programmen kommunizieren.

2.7.1 Konfigurationsdatei im ASCII-Format

Die Konfigurationsdatei enthält die wichtigsten Informationen für den Programmablauf, z. B. Daten zum Anwender, Pfad und Name der Eingabedatei sowie der Ausgabedateien und der Templatedatei für die Ausgabeerstellung (Maskendatei). Falls erforderlich können weitere Einträge für die Konfigurationsdatei definiert werden.

2.7.2 ASCII-Eingabedatei

In der Eingabedatei werden Aufbau und Struktur des Getriebemodells definiert sowie die Eingabewerte übergeben. Die Daten sind blockweise geordnet, wobei die Werte einem Bezeichner mittels „=“-Zeichen zugewiesen werden. Die Datei muss mit „$ Anfang“ beginnen. Das Ende der einzulesenden Daten ist mit „$ Ende“ erreicht. Als Beispiel sind aktuelle Ausgabedateien bestehender FVA-Programme heranzuziehen.

2.7.3 Zuordnungsdatei

Im Sinne einer Übersetzungstabelle zwischen ASCII-Eingabedatei und Rechenkern inkl. der Möglichkeit Defaultwerte und Ober-/Untergrenzen für jeden Parameter angeben zu können, ist die Zuordnungsdatei aufzubauen.

Die Zuordnungsdatei dient zur Entkoppelung von ASCII-Eingabedatei und Einlesebefehlen im Programm. In der Zuordnungsdatei wird für jeden Bezeichner in der Eingabedatei ein interner Bezeichner festgelegt, unter dessen Verwendung auf den entsprechenden Wert zugegriffen wird. Außerdem sind in der Zuordnungsdatei Defaultwerte und Wertebereich für jedes Datum angegeben.

2.7.4 Maskendatei

Mit der Maskendatei werden die Ausgabetexte in den Ergebnisdateien konfiguriert. Hiermit ist sicherzustellen, dass sämtliche Ausgabetexte unabhängig vom Programmcode (ohne erneutes Kompilieren und Linken) geändert werden können. Damit ist auch eine einfache Umsetzung einer mehrsprachigen Ausgabe möglich.

2.7.5 Schnittstellendatei

Die Schnittstellendatei dient zur programmbasierten Auswertung der Berechnungsergebnisse. Diese Datei kann von der FVA-Workbench oder anderen Programmsystemen verwendet werden, um im weiteren Rechenverlauf benötigte Daten aus vorhergehenden Berechnungen einzulesen. In die Schnittstellendatei sind alle Eingangsdaten sowie alle Berechnungsergebnisse des Programms aufzunehmen. Minimalumfang sind alle Daten, die dem Anwender in den ASCII-Textausgabedateien zur Verfügung gestellt werden.

Das bevorzugte Format der FVA-Workbench zum Datenaustausch ist die REX-Schnittstelle.

2.7.6 ASCII-Textausgabedateien

Die Berechnungsergebnisse sind in Form von Textdateien auszugeben. Es sind jeweils im Kopf der Programmname und das Datum der Berechnung sowie der Name der Berechnungsgröße, das Formelzeichen, der Wert und die Einheit anzugeben. Als Beispiel sind aktuelle Ausgabedateien bestehender FVA-Programme heranzuziehen.

2.7.7 Grafikausgabe

Die grafische Ausgabe wird in der Anforderungsliste festgelegt und erfolgt in Dateiform (z. B. LRZ (alt), XML, SVG). Die Ergebnisgrafiken können danach in einem Viewer betrachtet werden,

2.7.8 Errordatei

In die Errordatei sollten optional alle Fehlermeldungen umgeleitet werden können. Zusammen mit aussagekräftigen Kommentaren kann eine derartige Datei auch der Entwicklung zur Fehlersuche dienen. Für die Anwendung in der FVA-Workbench ist die MAX-Schnittstelle für die Ausgabe von Anmerkungen, Hinweisen und Fehlern zu verwenden.

2.7.9 Weitere Dateischnittstellen

Nach Bedarf sind Daten-Dateien, z. B. aus der Messung mit einer 3D-Koordinatenmessmaschine etc. umfassend zu dokumentieren.

2.8 Quelltext

2.8.1 Grundlegende Richtlinien für den Programmcode

Für die Erstellung von neuem Programmcode sind die Clean-Code-Regeln entsprechend der FVA-Definition anzuwenden. Die FVA-CCD-Richtlinie ist in ihrer aktuellen Fassung beim FVA Software-Kompetenzzentrum anzufragen [4].

2.8.2 Dokumentation im Quellcode

Die ausführliche Kommentierung des Quelltextes ist selbstverständlich.

Es ist ein Standard-Header am Anfang jeder Routine/jedes Objekts/etc. zu verwenden. Dabei kann auf das Dokumentationssystem der jeweiligen Programmiersprache zurückgegriffen werden:

- Java: JavaDoc
- C#.NET: Doxygen
- Fortran: Doxygen

Da sich die Formate der einzelnen Dokumentations-Generations-Systeme unterscheiden, sind nachfolgend nur die Informationen aufgeführt, die jeweils enthalten sein müssen / nicht enthalten sein sollen.

Informationen, die der Quellcode enthalten muss:

- Beschreibung der Funktion der Routine/des Objekts/etc.
- Beschreibung aller Variablen:
  i. Variablenname
  ii. Datentyp
  iii. Dimension
  iv. Kennzeichnung, ob Eingabe-/Ausgabewert oder beides
  v. Beschreibung
  vi. Erwartete Maßeinheit

Informationen, die der Quelltext nicht enthalten soll:

- Tagebuch-Kommentare (Wer hat wann was geändert)
  → Aufgabe des Quelltextverwaltungssystems
- Autor der Routine
  → Aufgabe des Quelltextverwaltungssystems

2.9 Dokumentation

Die zu implementierenden Funktionen und Gleichungen müssen definiert und dokumentiert werden.

Die Dokumentation muss einen theoretischen Grundlagenteil, eine Benutzeranleitung sowie eine Programmdokumentation enthalten. Folgende Hauptgliederungspunkte sollten enthalten sein:

Theoretische Grundlagen

- Überblick
- Entwicklungshistorie
- Theoretische Grundlagen
- Im Programm umgesetzte Formeln und Rechenmethoden

Benutzeranleitung

- Eingabemöglichkeiten
- Ergebnisausgaben
- Fehlerbehandlung

- Beispiele

Programmdokumentation

- Programmaufbau
- Struktur
- Schnittstellen

Unterlagen zum Workshop

- Einführung in die Theorie der Berechnungsgrundlagen
- erklärte Beispielrechnungen

Parametermanual (tabellarischer Anhang)

- Auflistung aller Parameter, die vom Programm gelesen werden
- Auflistung aller Parameter, die vom Programm ausgegeben werden
- Beschreibung des Parameters
    - Herkunft (Quelle: z. B. DIN, Konstruktionsmaß, physikalische Eigenschaft)
    - Zweck
    - Einfluss auf die Berechnung

Die einzelnen Abschnitte sollen folgende Inhalte beschreiben:

Theoretische Grundlagen:

Der Überblick enthält eine Einführung in die Berechnungssoftware (Wozu dient die Berechnung? Was kann berechnet werden? Wie kann berechnet werden? Wo sind die Grenzen der Berechnung?)

Der Umfang dieses Teils sollte maximal zwei Seiten betragen (ähnlich dem Info-Blatt).

Die Entwicklungshistorie soll einen knappen Überblick über den Leistungsumfang früherer Programmversionen geben und stellt die Release-Dokumentation dar.

Die theoretischen Grundlagen für die Berechnung sind zu dokumentieren. Die im fertigen Programm tatsächlich verfügbaren Rechenmethoden sind darzustellen.

Benutzeranleitung:

Die Eingabephilosophie und -möglichkeiten, das typische Vorgehen bei der Definition der Berechnungsaufgabe und eine Beschreibung der Besonderheiten, wie Materialdatenbanken, Zusatzdateien etc., sind hier zu dokumentieren. Wird das Programm in die FVA-Workbench integriert, muss eine Zuordnungstabelle zwischen den Eingabegrößen in der FVA-Workbench und den Eingabevariablen der REXS-Eingabedatei zur Verfügung gestellt werden.

In der Benutzeranleitung sollen die Beispiele, die den Testern (und Endanwendern) zur Verfügung stehen, beschrieben werden. Es ist die konkrete Anwendungssituation, die Parameter (Bedeutung, Quelle, Gültigkeit) sowie die Interpretation und Diskussion der Ergebnisse und deren praktische Bedeutung für das berechnete System zu beschreiben.

Programmdokumentation:

Die Programmdokumentation ist die Bedienungsanleitung für den Programmierer.

Sie beinhaltet weniger die Dokumente zur Entwicklung selbst, sondern behandelt die Nutzungsmöglichkeiten der wieder verwendbaren Funktionen in den Anwendungssystemen, die den Softwareentwicklern zur Verfügung stehen.

Der Programmablauf ist Anhand von Diagrammen zu beschreiben. Falls erforderlich, kann die Programmstruktur der Software getrennt dokumentiert werden,.

Im Programm enthaltene Subroutinen und Variablen sind nach Name und Aufgabe zu dokumentieren. Die Ein- und Ausgabeparameter müssen eindeutig definiert sein. Es kann auf umfangreich dokumentierte Quelldateien verwiesen werden bzw. können die Header der Quelltextdateien zur Dokumentation ausgedruckt werden.

Die Dokumentation der Programmschnittstellen bezieht sich auf alle Ein- und Ausgabedateien des Programms. Aufbau und Inhalt sind systematisch zu dokumentieren.

2.10 Konvention zur Festlegung der Versionsnummer

Generell sind alle Programme mit numerischen Versionsnummern auszustatten. Die Nummerierung unterliegt folgenden Konventionen:

<Programmname>AA.BB.CC.EE

AA = Major-Release

Diese ändern sich nur, wenn signifikante Änderungen oder Erweiterungen am Programm vorgenommen werden (der Nutzwert der Software erhöht sich). Der Wertebereich für AA ist 1 bis 99. Die Versionsnummer des Major-Release ist konsequent immer nur um einen Zähler zu erhöhen. Die Null ist bei einstelligen Versionsnummern wegzulassen.

Beispiel: STplus 4.01 (STplus 04.01 hingegen ist nicht zulässig)

Programme, die nur in der Majorversion verfügbar sind, sind wie folgt zu nummerieren:

Beispiel: PressFit 1.0

BB = Minor-Release

Diese Nummer ändert sich typischerweise, wenn an dem Programm Bugfixes oder kleinere Änderungen und/oder Erweiterungen vorgenommen werden (der Nutzwert der Software erhöht sich kaum). Der Wertebereich für BB ist 1 bis 9. Diese muss aber nicht bis zur 9 weitergeführt werden, um einen Wechsel der Major-Release-Nummer durchzuführen.

Beispiel: BECAL 3.05 auf Becal 4.0 ist möglich.

CC = Versionierung

Diese Nummer dient zur Versionierung bei Behebung von Fehlern und/oder kleineren Korrekturen in Modulen oder Funktionen. Der Funktionsumfang der Software wird nicht geändert. Der Wertebereich für CC ist 1 bis 9.

EE = interne Versionsnummern

Es bietet sich an, anstatt einer rein numerischen Darstellung die Versionierung nach dem Erstellungsdatum zu klassifizieren.

Beispiel: BECAL 3.05_Build20050412. Es handelt sich hierbei um das Programm, das am 12.04.2005 erstellt wurde. In der Regel sollten diese Nummern nur zum internen Gebrauch oder zur Kenntlichmachung in der Beta-Phase verwendet werden. Es kann jedoch sinnvoll sein, dass diese Nummer in Programmausgaben mit ausgeben wird. Somit kann identifiziert werden, von welcher Version (und Binär-Datei) der Ausdruck stammt.

2.11 Dokumentation der Änderungshistorie

Die Änderungshistorie ist sowohl innerhalb eines Vorhabens als auch über mehrere Vorhaben hinweg darzustellen.
Innerhalb eines Vorhabens soll zum Beta-Workshop und zum Abnahmeworkshop die Abarbeitung von Fehlern (Bugliste) vorgestellt werden.

Der Funktionszuwachs über mehrere Vorhaben hinweg ist ebenfalls zu dokumentieren. Enthalten sein müssen:

- Versionsnummer und Release-Datum,
- neue Funktionen,
- Erweiterungen,
- beseitigte Fehler.

Das Format der Dokumentation ist frei wählbar, muss aber menschenlesbar sein. Die Dokumentation ist im obersten Verzeichnis der Programminstallation abzulegen.

2.12 Archivierung

Während der laufenden Entwicklung wird ein Versionsverwaltungssystem verwendet (z. B. GIT).

Die Programmversion des Abnahmeworkshops wird dem FVA-Software-Kompetenzzentrum zur langfristigen Archivierung übergeben.

3 Anhang

3.1 Anforderungsliste

Die Anforderungsliste legt zu Beginn des Vorhabens begleitend zum wissenschaftlichen Forschungsinhalt im Antrag die Anforderungen an die Softwareentwicklung fest.

Anforderungsprofil: Software innerhalb von Forschung

Forschungsvereinigung Antriebstechnik e.V.

Funktionale Anforderungen

1. Benennung:

- Name: _____
- Version: _____

2. Programmformen:

- ☐ Stand -Alone
- ☐ Konsolenprogramm mit externer Abhängigkeit
- ☐ Programmbibliothek
- ☐ Skript
- ☐ Anderes: _____

3. Support nach der Laufzeit:

- ☐ durch das forschende Institut
- ☐ durch Folgevorhaben
- ☐ durch anderes Institut / Mitgliedsfirma
- ☐ durch Community
  (Erstellung einer Nutzerliste und Ablage ALLER Daten im Softwarekompetenzzentrum)
- ☐ durch FVA GmbH (Softwarekompetenzzentrum)

4. Schnittstellen:

- ☐ REXS Eingabe / Ausgabe
- ☐ MAXS
- ☐ ASCII Eingabe / Ausgabe
- ☐ ASCII (Neutrale Schnittstelle)
- ☐ Fehlercode basierte Ausgabe
- ☐ Anderes: _____
- ☐ Schnittstellen an- und ausschaltbar
- ☐ Umleitung auf Standard IN/OUT

5. Grafik / CAD Schnittstelle:

- ☐ PDF
- ☐ JPG
- ☐ Tiff
- ☐ Ausgabe als CAD-Punktewolke
- ☐ Step-Datei
- ☐ Anderes Format: _____

6. Kompatibilität:

- ☐ Abwärtskompatibel (Alte Datensätze können verarbeitet werden)

7. Performance:

- ☐ Neues Feature darf Programm nicht verlangsamen
- ☐ Es ist zulässig, dass durch neue Features die Performance zurück geht

8. Testfälle:

- ☐ Testfälle der FVA GmbH
- ☐ Andere ‚offene' Testfälle: (werden in die Testdatenbank aufgenommen)
- ☐ Andere ‚Firmeninterne' Testfälle: (verbleiben im Institut)

9. Zeitpunkte für Q-Gates:

- ☐ Zeitpunkt für Alpha Workshop: _____ Monate
  (in Monaten nach Start des Forschungsprojektes)
- ☐ Zeitpunkt für Beta Workshop: _____ Monate
  (in Monaten nach Start des Forschungsprojektes)
- ☐ Zeitpunkt für Abnahme Workshop: _____ Monate
  (in Monaten nach Start des Forschungsprojektes)

10. Art des Workshops:

Alpha:
- ☐ Umlauf
- ☐ Webmeeting
- ☐ Präsenztermin

Beta:
- ☐ Umlauf
- ☐ Webmeeting
- ☐ Präsenztermin

Abnahme:
- ☐ Präsenztermin
  (Ziel ist der Präsenztermin, bei einfachsten Programmen können auch die anderen Optionen gewählt werden)
- ☐ *Umlauf*
- ☐ *Webmeeting*

Nicht funktionale Anforderung

2. Bevorzugte Programmiersprache:

- ☐ **Fortran** nach ISO Standard
- ☐ **Java**
- ☐ **C** nach ANSI Standard
- ☐ **C ++** nach ISO Standard

} portierbare Sprache

- ☐ Andere: _____

3. Implementierung in die Workbench erwünscht:

- ☐ Ja (Bedingt zeitgleichen Antrag auf ein Transferprojekt)
- ☐ Nein

4. Verwendete Programmbibliotheken:

- ☐ FVA: _____
- ☐ Andere: _____

5. Verwendung von Modulen:

- ☐ Nein
- ☐ Ja: _____
  (welche Operationen sollen von anderen Programmen / Programmteilen übernommen werden)

6. Dokumentation:

- ☐ Compiler inkl. Versionen und Einstellungen
- ☐ Anleitung zum Programm
- ☐ Andere: _____

7. Startpunkt für Folgevorhaben:

- ☐ Ausgangsprogramm: _____
- ☐ Version: _____
- ☐ Modulversion: _____

8. Strategie zur Verbreitung:

- ☐ FVA – intern
- ☐ Normenausschuss
- ☐ Workbench Extern (Zugang für Nicht-FVA-Mitglieder)
- ☐ International
- ☐ Andere: _____

3.2 Software-Entwicklungsstufen Methodenträger, Forschungssoftware und Anwendungssoftware zur Orientierung

Die in früheren Versionen dieser Richtlinie eingeführten Software-Entwicklungsstufen Methodenträger, Forschungssoftware und Anwendungssoftware sind hier zur Orientierung mit ihren inhaltlichen Beschreibungen noch einmal aufgeführt. Die Anforderungsliste kann so ausgeprägt werden, dass sich ein Anforderungsprofil ergibt, das inhaltlich einer der nachfolgend aufgeführten Software-Entwicklungsstufen entspricht.

3.2.1 Methodenträger

Ziel des Methodenträgers ist die automatisierte Berechnung zur eingeschränkten Anwendung der entwickelten Berechnungsmethoden eines Forschungsvorhabens. Hiermit soll es möglich sein, neue Berechnungsansätze zu validieren, bevor in aufwendige Softwareentwicklung investiert wird. Methodenträger ermöglichen allein die industrielle Überprüfung neuer wissenschaftlicher Ansätze. Die Anforderungen an die Umsetzung werden vom projektbegleitenden Ausschuss (Arbeitsgruppe) definiert. Der Nutzerkreis ist auf die Mitglieder der Arbeitsgruppe beschränkt. Eine weitere Verwendung des Methodenträgers, z. B. in der FVA-Workbench, ist nicht vorgesehen. Methodenträger können, z. B. in einem Folgevorhaben, nicht weiterentwickelt werden. Soll die Berechnungsfunktionalität eines Methodenträgers, z. B. in einem Folgevorhaben, verändert oder erweitert werden, so ist der Methodenträger in Forschungs- oder Anwendungssoftware zu überführen. Die Prüfung, die Freigabe und die Sicherstellung der Funktionalität und der Ergebnisse eines Methodenträgers liegen in der Verantwortung des Projektleiters (Industrie). Ist der Projektleiter nicht mehr in der FVA aktiv, so fällt dem Projektleiter der AG diese Aufgabe zu. Ist dieser auch nicht mehr in der FVA aktiv, so fällt diese Aufgabe an den Obmann des Arbeitskreises.

3.2.2 Forschungssoftware

Unter Forschungssoftware versteht man bestehende Programme, die an Forschungsstellen im Rahmen von Forschungsprojekten entwickelt wurden. Der Nutzerkreis sind alle FVA-Mitglieder und Forschungsstellen. Im Rahmen des Vorhabens werden Testfälle erstellt, welche die Funktionsfähigkeit des Programmes nachweisen und später als Referenz für die weiteren Entwicklungen dienen. Nach Abschluss des Vorhabens werden sämtliche entwicklungsrelevanten Dokumente, die Forschungssoftware selbst in Form eines ausführbaren Programmes und als Quellcode dem FVA-Software-Kompetenzzentrum übergeben.

Für die Pflege und Anwenderunterstützung ist die erstellende Forschungsstelle zuständig.

Eine Forschungssoftware unterliegt denselben Qualitätsbedingungen wie eine Anwendungssoftware. Dies bezieht sich vor allem auf die Aspekte Zuverlässigkeit, Fehlerfreiheit und das Vorhandensein einer ausreichenden Anzahl relevanter Testfälle. Vor Veröffentlichung von Forschungssoftware werden die gelieferten Dateien vom FVA-Software-Kompetenzzentrum hierauf geprüft.

Lediglich Aspekte wie die Entwicklung nach Clean-Code-Paradigmen oder die Verwendung von Unit-Tests sind hier nicht zwingend vorgeschrieben (können aber im Einzelfall hilfreich sein).

Die Prüfung, die Freigabe und die Sicherstellung der Funktionalität und der Ergebnisse von Forschungssoftware liegen in der Verantwortung des Projektleiters (Industrie). Ist der Projektleiter nicht mehr in der FVA aktiv, so fällt dem Projektleiter der AG diese Aufgabe zu. Ist dieser auch nicht mehr in der FVA aktiv, so fällt diese Aufgabe an den Obmann des Arbeitskreises.

3.2.3 Anwendungssoftware

Ziel der FVA ist es, den Unternehmen die Forschungsergebnisse in Form einer in die FVA-Workbench integrierten Anwendungssoftware zur Verfügung zu stellen. Die Anwendungssoftware soll im Umfang der FVA-Workbench auf alle Getriebe in der industriellen Praxis anwendbar sein.

Im Rahmen eines Forschungsprojektes wird ein Methodenträger oder eine Forschungssoftware an der Forschungsstelle erstellt. Wird die umgesetzte Berechnungsfunktionalität von den Industrievertretern als für die industrielle Praxis sinnvoll erachtet, kann die Überführung in Anwendungssoftware beantragt werden. Eine Überführung der Quelltexte unter Anwendung der erweiterten Anforderungen aus [4] in eine Anwendungssoftware erfolgt in Zusammenarbeit mit der FVA Software & Service.

Die FVA Software & Service übernimmt die weitere Bearbeitung, Wartung und Pflege der Anwendungssoftware.

3.2.3.1 Anforderungen an die Entwicklung von Forschungssoftware

Forschungssoftware, in der Regel an Forschungsstellen im Rahmen von Vorhaben erstellt, ist typischerweise charakterisiert durch Stand-Alone-Programme, die von allen FVA-Mitgliedern und Forschungsstellen genutzt werden. Es gelten grundsätzlich die Vorgaben aus den Kapiteln 1 und 2 und damit dieselben Anforderungen wie für die Anwendungssoftware, abzüglich der Vorgaben zur Beachtung der Clean Code Paradigmen bei der Programmierung (vgl. Anlage [4]).

3.3 FVA-Workbench-Integration

Die Anforderung, einen modularen Softwarebaustein in die FVA-Workbench zu integrieren, ist in der Anforderungsliste zu definieren. Bei der Integration von Rechenmodulen in die FVA-Workbench sind Randbedingungen zu beachten, die mit dem FVA-Software-Kompetenzzentrum abzustimmen sind.

4 Ergänzende Dokumente

Folgende Dateien sind in der aktuellen Version diesem Dokument mit beigefügt bzw. können zusätzlich bei der FVA-Geschäftsstelle angefragt werden:

[1] Vorlage Datenblatt Rechenkern-Release

[2] Formular zur offiziellen Abnahme des Programms durch die Arbeitsgruppe

[3] Vorlage zur Erstellung von User-Stories

[4] FVA-CCD-Richtlinie

[5] Leitfaden FVA e.V.