PROJEKTHANDBUCH

Entwicklung einer Firefox-Extension für BCI-Hardware

[FBCI]

Version 0.3

Projektleiter: Andreas Kirchner

Datum: 02.05.2009

Inhalt

Änderungsverzeichnis

Versions-nummer	Änderung	Ersteller
0.1	• Kurzbeschreibung • Projektstrukturplan incl. Ausführlicher Phasenbeschreibung • Projektbalkenplan	Kirchner
0.2	• Erläuterung des Projektbalkenplans • 3 AP spezifiziert	Kirchner
0.3	• Projektumweltanalyse	Kirchner

Projektpläne

1.1 Kurzbeschreibung des Projekts

Vorgeschichte / Ist-Stand

Die Bachelorarbeit im Zusammenhang mit dem Projektpraktikum (nachfolgend das Projekt genannt) beschäftigt sich mit Softwareentwicklung im Themenkreis „Brain Computer Interfaces“ (kurz: BCI). Grob kann man zwischen invasiven und nicht-invasiven Schnittstellen zwischen Gehirn und Computer unterscheiden. Im Projekt steht uns (vom Institut für Distributed and Multimedia Systems) nicht-invasive BCI-Hardware zur Verfügung (http://www.gtec.at/products/g.Accessories/Cap.htm). Die Benutzerin setzt eine Kappe mit EEG-Elektroden auf, welche ihre Gehirnströme misst. Die gemessenen Signale werden durch einen Biosignal-Verstärker (http://www.gtec.at/products/g.USBamp/gUSBamp.htm) zum USB-Port des Computers weitergeleitet. Der Hersteller GTec bietet zu seiner Hardware ein Trainingsprogramm und ein Simulink-Modul an, das zusammen genommen eine Echtzeit-Klassifikation der Gehirnaktivitäten des motorischen Cortex zulässt. Nicht nur erzeugt die willentliche Bewegung der Arme und Beine eine Aktivität im Motorcortex; auch *Vorstellungen* von Bewegungen der Gliedmaßen (linke Hand, rechte Hand, Bein) bringen messbare Aktivität hervor. Dadurch wird Benutzerinteraktion aufgrund von Bewegungsvorstellungen möglich, was genau der Intention von BCI entspricht.

Es lassen sich in der vom Hersteller mitgelieferten Software grob zwei Modi unterscheiden:

- Im Trainingsmodus werden die Parameter angepasst, die eine Korrelation zwischen der Vorstellung der Bewegung und den gemessenen Gehirnaktivitäten erlauben. Die Parameter sind für jede Person mehr oder weniger verschieden. Zu bestimmten Zeitpunkten wird die Userin aufgefordert, an eine bestimmte Bewegung (z.B. des linken Armes) zu denken. Aufgrund des visuellen Feedbacks, das die Software gibt, lernt die Userin, welche Gedanken für die Klassifikation effizient sind. Andererseits passt die Software ihre Parameter auf die individuellen Muster der Gehirnaktivität an, um die Klassifikation zu optimieren.
- Im asynchronen Modus, der nach dem Trainingsmodus verwendet wird, kann die Userin zu beliebigen Zeitpunkten an beliebige Bewegungen (Arm links, Arm rechts, Bein) denken. Durch ein Software-Modul, das in Simulink (ein Simulationstool in Matlab) integriert ist, wird eine Echtzeitklassifikation möglich, die für unterschiedliche Anwendungen verwendet werden kann.

Problematik / Motivation

Die Motivation, auf Basis des obigen BCI-Systems unser Projekt zu starten war, dass die bis jetzt entwickelten Anwendungen...

- sich auf *wenige Demonstrationsbeispiele* im Sinne von „Die Klassifikation funktioniert“ beschränken.
- „*nur*“ für eine *kleine Zielgruppe* (akademisches Personal oder Personen mit dem „Locked-In-Syndrom“) entwickelt wurden.
- *keine einfache Schnittstelle zu Web-Applikationen* erlauben. Man kann die Hardware nicht ohne Weiteres zur Steuerung von Web-Applikationen nutzen.

Unser Projektziel ist die Entwicklung einer Extension für Firefox, die auf Basis der bestehenden BCI-Systems eine Steuerung von Web-Applikationen zulässt. Dabei sind zwei Benutzermodi vorgesehen:

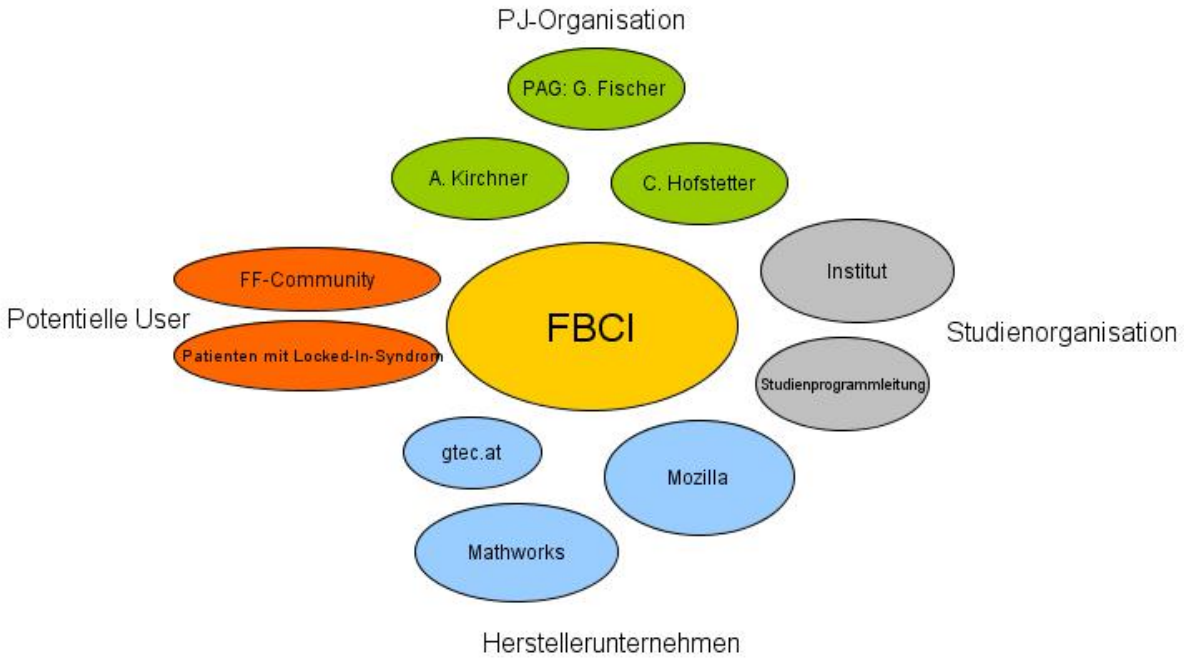
- Eine *generische Applikations-unabhängige Navigation* sorgt dafür, dass unabhängig davon, welche Web-Applikation verwendet wird, ein minimales Set an Navigationsmöglichkeiten verwendet werden kann. Der Vorteil ist, dass der bestehende Code der Applikation nicht geändert werden muss.

- Eine *spezifisch für die Applikation angepasster Modus* erlaubt den Entwicklern von Web-Applikationen eine maximale Freiheit, wofür sie die Klassifikation der Gehirnaktivitäten in der Programmlogik verwenden.

Die Entwicklung soll durch die Bachelorarbeit dokumentiert werden, was zum zweiten Projektziel führt: Das Abschließen des Bachelorstudiums durch Einreichen der Bachelorarbeit. Dabei sind innerhalb des Projekts (im Rahmen der Projektkoordination) die formalen und inhaltlichen Kriterien mit dem Betreuer und der Studienprogrammleitung zu klären.

1.2 Projektumwelt-Analyse

PROJEKTUMWELTEN-GRAPHIK



PROJEKTUMWELTEN-BEZIEHUNGEN

Umwelten	Beziehung (Potential/Konflikt)	Maßnahmen	PSP – Code
Studienprogrammleitung	Konflikt: Bakk-Arbeit muss pünktlich zu Semesterende abgegeben werden	Informationen über die Einreichfrist einholen bzw. ggf. Aufschub erwirken	1.1.3 (PJ-Koordination)
gtec.at	Potential: Das Unternehmen unterstützt das Projekt durch Wissen oder Ressourcen	Kontaktaufnahme mit Projektbeschreibung und Ansuchen um Hilfe bzgl. undokumentierter Informationen	1.1.3 (PJ-Koordination)
Institut für Distributed and Multimedia Systems	Potential: BCI-Hardware wird zum Testen zur Verfügung gestellt	Zu Beginn einen Projektüberblick geben für den BCI-HW-Zuständigen und Benutzungsmodalitäten klären	1.1.2 (PJ-Start)

FF-Community	Potential: Support, Weiterentwicklung und Nutzung der Extension	Im Zuge des ersten Tests erfolgt die Veröffentlichung der Extension (wobei der HW-spezifische Teil noch ausgespart ist) für die FF-Community mit der Bitte, Feedback zu geben und die Funktionalität zu testen Im Zuge der finalisierenden Fehlerbehebung erfolgt ebenfalls nochmals eine Veröffentlichung, wobei der Closed-Source-Teil aus rechtlichen Gründen ausgespart werden muss (Sprich: man braucht die BCI-HW und muss Matlab kaufen, um den HW-spezifischen Teil zu benutzen). Jedoch können der Extension zusätzliche Module hinzugefügt werden, die andere BCI-HW-Schnittstellen implementiert	1.4.4 (Funktionalität der Extension testen) 1.6.3 (Letzte Fehler beheben)
PAG: G. Fischer	Potential: Feedback bzgl. Medizinischer Constraints und Biosignalverarbeitung	Durch regelmäßige Rücksprache mit dem PAG werden dessen Kompetenzen im medizinischen Bereich und der Biosignalverarbeitung verwendet.	1.1.3 (PJ-Koordination)

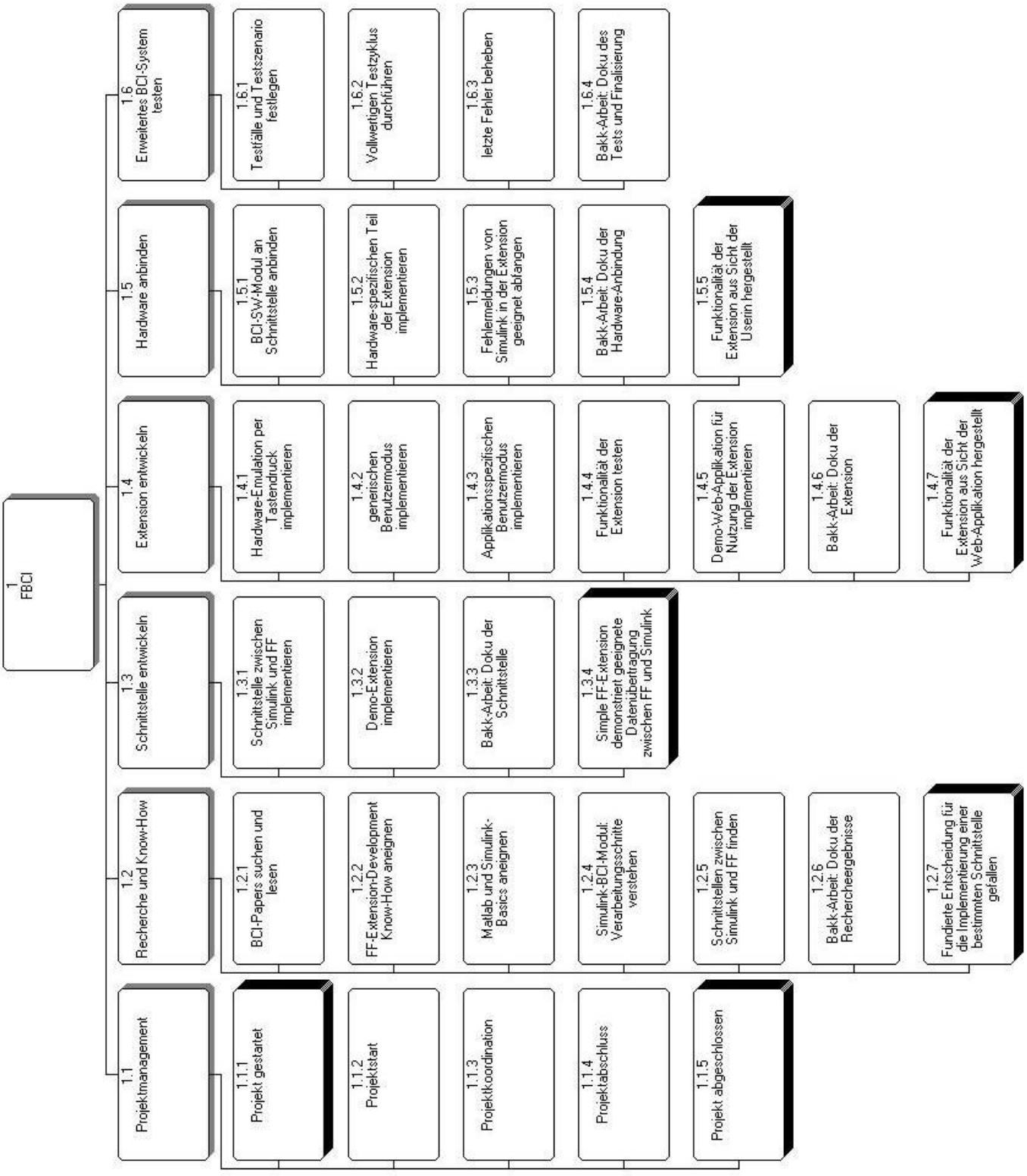
Die relevanten Umwelten der PUA halten sich in Grenzen, da die Projektorganisation aus nur zwei Mitarbeitern + Projektauftraggeber (=Betreuer) besteht.

Die Umweltgruppe „Studienorganisation“ ist für den Erfolg des Projekts relevant, da einerseits die Hardware von einem Institut der Universität Wien zur Verfügung gestellt wird, andererseits die Deadlines für das Einreichen der Bachelorarbeit von der Studienprogrammleitung abhängen.

Die Herstellerunternehmen spielen im Support-Bereich eine Rolle. Es könnte sein, dass während des Umgangs mit den Produkten des Unternehmens Fragen auftauchen, die eine Unterstützung der Hersteller erfordern (z.B. wenn es nicht-dokumentierte Funktionalitäten gibt, oder unbekannte Bugs auftauchen, etc.). Für gtec.at, dem BCI-Hardware gilt im Speziellen, dass die Entwicklung der Extension für das Unternehmen selbst relevant sein könnte, was im Laufe des Projekts zu klären sein wird.

Potentielle User können für die finale Projektphase, was das Finden von Bugs oder das Aufzeigen von fehlenden Funktionalitäten betrifft, eine Rolle spielen. Ein gesellschaftlich relevanter Projektnutzen könnte sich für Patienten mit Locked-In-Syndrom ergeben, z.B. durch eine Kooperation mit Krankenhäusern; dies wird ggf. in einer Nachprojektphase erörtert.

1.3 Projektstrukturplan



1.3.1 Beschreibung des PSP:

Das Projekt gliedert sich in fünf Phasen auf. In jeder Phase gibt es ein "Bakk-Arbeit"-Arbeitspaket, in dem die Ergebnisse der Phase schriftlich festgehalten werden. Aus den Dokumenten der einzelnen "Bakk-Arbeit"-AP resultiert die einzureichende Bachelor-Arbeit.

Im Folgenden werden die Ziele der einzelnen Phasen und ihre Ergebnisse kurz vorgestellt:

1.2 Recherche und Know-How:
Die einzelnen APs sollen...:
(a) den Forschungsstand von BCI-Systemen allgemein erkunden (1.2.1),
(b) die grobe Funktionsweise des eingesetzten BCI-Systems feststellen (1.2.4),
(c) die Möglichkeiten von in Frage kommenden Technologien und Tools erkunden (Matlab, Simulink, FF, XML-RPC, JavaScript, XUL,...) (1.2.2, 1.2.3)
(d) Möglichkeiten, den Datentransfer zwischen Simulink und Firefox (FF) zu bewerkstelligen, herausarbeiten und abwägen (1.2.5).

Als Endergebnis dieser Phase soll eine für den weiteren Projektverlauf fundamentale Entscheidung getroffen werden: Wie wird der Datentransfer zwischen Simulink und Firefox bewerkstelligt; wie wird diese Schnittstelle implementiert? Was waren die Alternativen und warum ist die Entscheidung zugunsten dieses Weges gefallen. (Meilenstein: 1.2.7)

1.3. Schnittstelle entwickeln:
Aufbauend auf 1.2.7 soll die Schnittstelle entwickelt werden (1.3.1). Um die Leistungsfähigkeit der Schnittstelle zu demonstrieren, wird eine kleine Firefox-Extension implementiert (1.3.2). Das Ergebnis dieser Phase ist der Source-Code der Schnittstelle und der Demo-Firefox-Extension, die zusammengenommen den kontinuierlichen Datentransfer zwischen FF und Simulink demonstrieren sollen. (1.3.4)

1.4. Extension entwickeln:
Die Phasen bis jetzt waren Vorbereitungen auf das eigentliche Ziel des Projekts: Eine FF-Extension zur Steuerung von Web-Applikationen durch Gedanken (genauer: Vorstellungen von Gliedmaßen-Bewegungen). Um die Extension sinnvoll zu entwickeln (Verbrauch der Leitpaste für die Sensoren; die Hardware ist nur am Institut verfügbar), wird zunächst der Hardware-abhängige Teil emuliert. Das heißt genauer, die Vorstellungen der drei Bewegungen werden durch das Drücken von drei Tasten ersetzt (1.4.1). Nun werden die beiden Benutzermodi implementiert (1.4.2, 1.4.3; siehe Projektbeschreibung). Vermittels der Hardware-Emulation wird parallel und im Anschluss die Funktionalität der Extension getestet (1.4.4). Für den Applikations-spezifischen Benutzermodus ist die Entwicklung einer Demo-Web-Applikation vorgesehen (1.4.5), um die Möglichkeiten der Extension zu demonstrieren (und die FF-Entwickler-Community auf den Geschmack zu bringen). Das Ende dieser Phase ist dann erreicht, wenn aus der Sicht der Web-Applikation die Funktionalität voll erfüllt ist (1.4.7). Denn sie kann nicht unterscheiden, ob die Events durch Tastendruck oder Gedanken ausgelöst wurden.

1.5. Hardware anbinden
Der in der dritten Phase suspendierte Teil der Hardware-Anbindung ist Ziel dieser Phase. Dieses Vorgehen, zunächst den Hardware-unabhängigen Code (Schnittstelle, Großteil der Extension) zu produzieren, hat den Vorteil, dass man sich anschließend mit allen Ressourcen auf die Kommunikation mit der Hardware konzentrieren kann. Da wir das Institut nicht ständig belagern können, muss die Hardware zeit-effizient angebunden werden. Dafür ist dieses Vorgehen genau richtig.

Die Ziele dieser Phase sind im Einzelnen:

(a) Das vom Hersteller mitgelieferte Simulink-Modul wird an die in der zweiten Phase implementierte Schnittstelle angebunden (1.5.1).
(b) Alle Hardware-relevanten Teile der Extension, sowie geeignete Reaktionen auf von Simulink oder der Hardware produzierte Fehlermeldungen implementieren (1.5.2, 1.5.3). So kann man geeignete Mechanismen entwickeln, die feststellen, ob die von der Extension unabhängige Trainingsphase für die Userin bereits durchgeführt wurde und anbieten, die Trainingsapplikation zu starten.

Das Ende dieser Phase ist erreicht, wenn alle festgelegten Funktionalitäten der Extension implementiert sind (1.5.5). Damit ist bis auf kleine Fehlerbehebungen die Softwareentwicklung abgeschlossen.

1.6. Erweitertes BCI-System testen

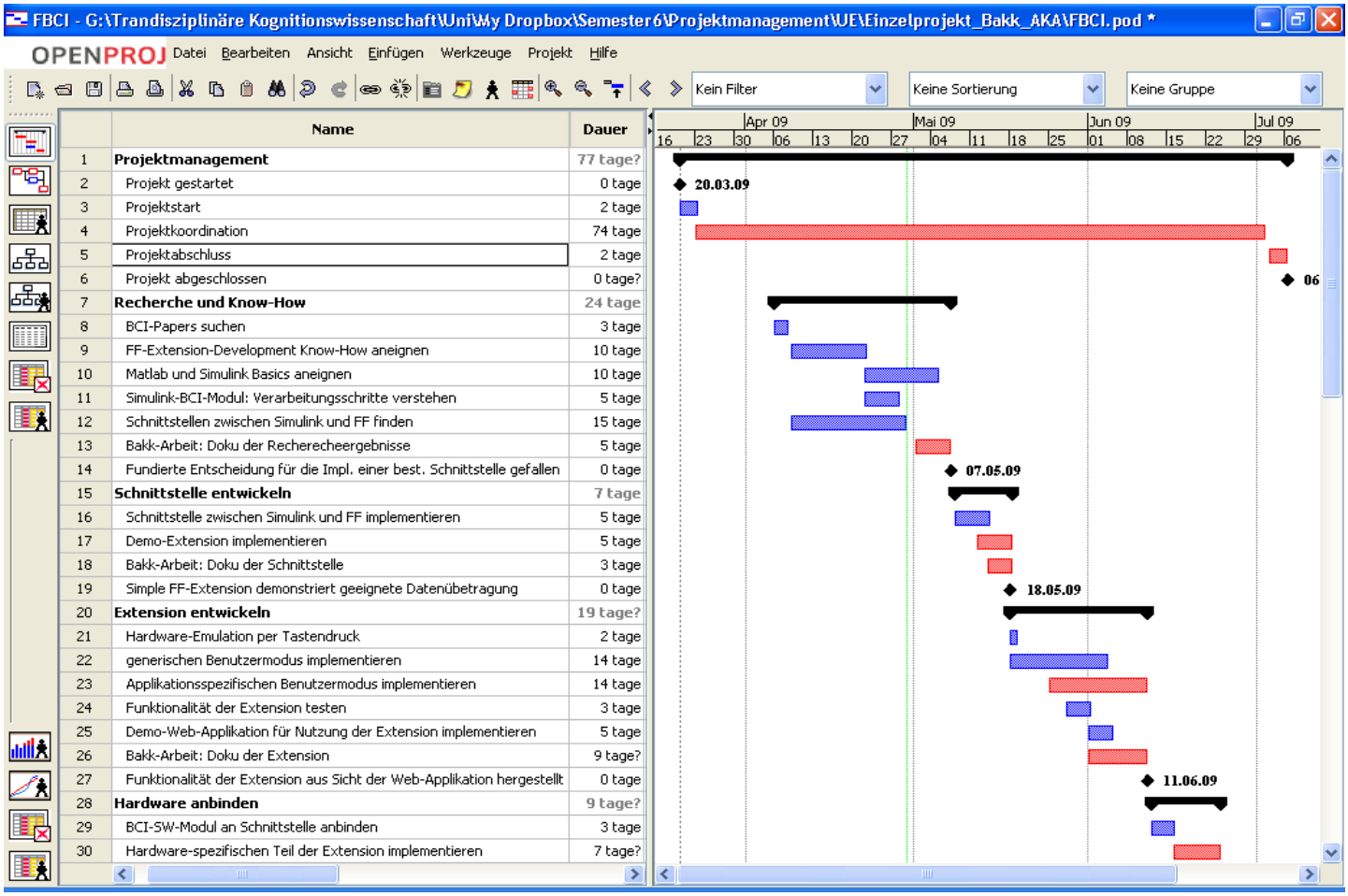
Auf Basis des vorliegenden Codes werden nun Testfälle formuliert und getestet. (1.6.1, 1.6.2) Nicht erwünschtes Verhalten wird entsprechend korrigiert (1.6.5). Mit der Integration und Überarbeitung der schriftlichen Dokumente aller Phasen zu einer Bachelor-Arbeit (1.6.4) endet das Projekt.

1.4 Arbeitspaket-Spezifikationen

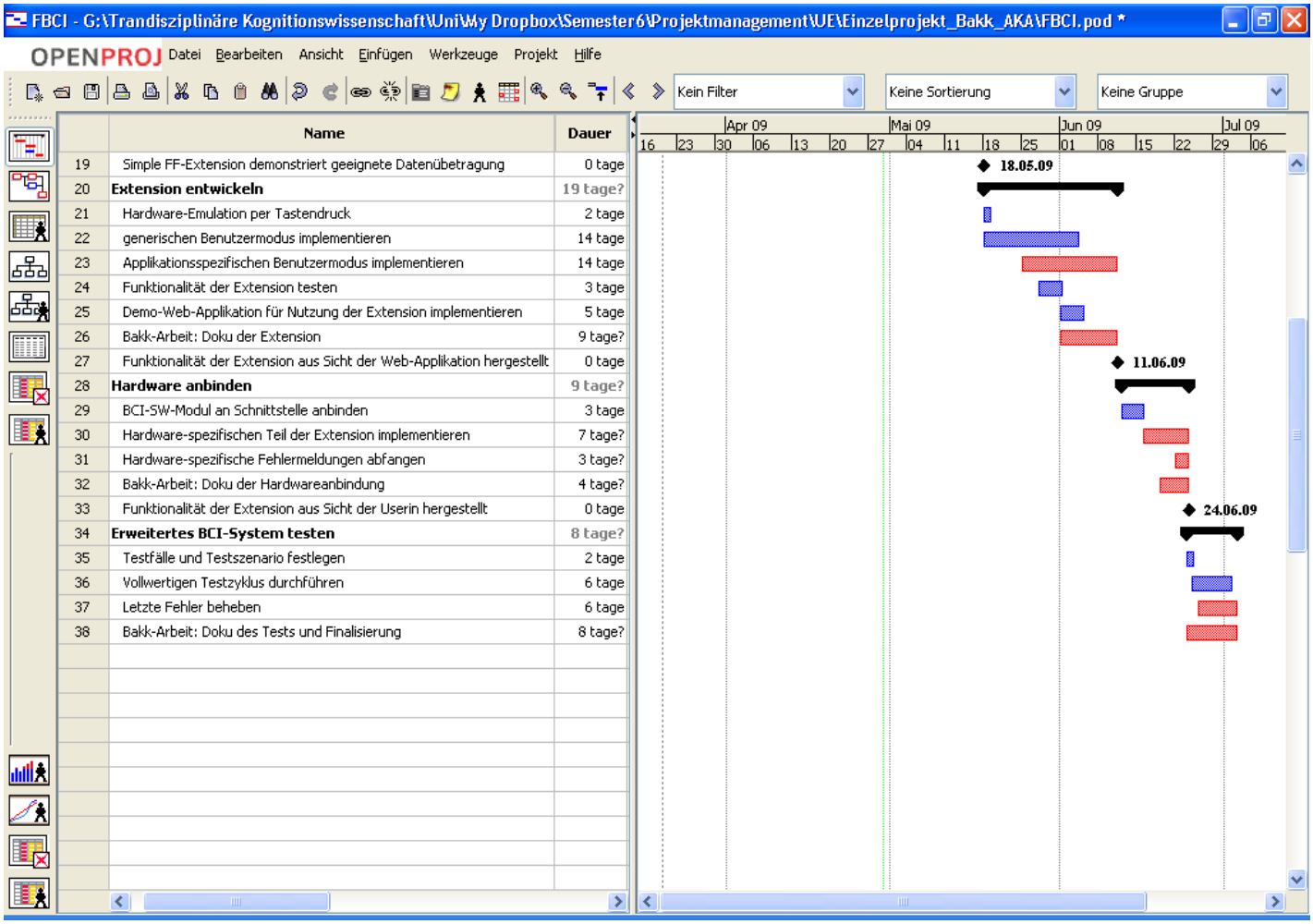
ARBEITSPAKET-SPEZIFIKATIONEN	
1.3.1 Schnittstelle zwischen FF und Simulink implementieren	AP-Inhalt: • Mit der in 1.2.7 entschiedenen Möglichkeit / Technologie, Source-Code entwickeln, der die Datenübertragung zwischen Matlab Simulink und Firefox realisiert • Es ist darauf zu achten, dass in Firefox die in Simulink generierten Daten, unmittelbar (< 100ms) zur Verfügung stehen, damit für die Userin ein unmittelbares Feedback und damit ein schneller Lernprozess möglich ist • Nachdem der erste Entwurf des simulink-seitigen Teils der Schnittstelle implementiert ist, erfolgt die Abstimmung mit dem AP 1.3.2 (Demo-Extension implementieren).
	AP-Nicht-Inhalte • Die Anbindung des BCI-Klassifikationsmoduls an die Schnittstelle. Hier geht es ausschließlich um die Datenübertragung zwischen den beiden Tools • Details der Ausformung der Demo-FF-Extension sollen nicht hier, sondern in AP 1.3.2 erfolgen. Hier soll als E2 nur ein Code-Stück erzeugt werden, das zeigt, wie man auf den Datenstrom von Simulink zugreift
	AP-Ergebnisse • E1: erster Entwurf des Source-Code des Simulink-Moduls, der erste Interaktionen mit dem Client (dem FF-Code-Teil) erlaubt • E2: Code-Teil, der in eine FF-Extension integriert werden kann. Dieser erlaubt Interaktion mit dem Server (dem Simulink-Teil) • E3: Funktionierende Schnittstelle, die durch die in AP 1.3.2 bereits teilweise entwickelte Extension demonstriert wird (siehe Balkenplan)
	AP-Leistungsfortschrittsmessung • Sobald E1 vorliegt: 40% • Sobald E2 vorliegt: 80% • Sobald E3 vorliegt: 100%
1.4.2 generischen Benutzermodus implementieren	AP-Inhalt: • Die Ergebnisse aus 1.4.1 (Emulation der HW durch Tastendruck) werden integriert, und die festgelegte Funktionalität des generischen Benutzermodus wird implementiert • Es ist darauf zu achten, dass die Ergebnisse aus 1.4.1 als unabhängiges Modul implementiert werden, das in den Einstellungen der Extension neben dem zweiten BCI-Hardware-abhängigen Modul (das in 1.5.2 implementiert wird), ausgewählt werden kann. • Es ist in der Eingangsphase des APs darauf zu achten, dass zunächst ein für beide Benutzermodi gültiges Framework der Extension implementiert wird, denn nach einigen Tagen beginnt parallel dazu die Entwicklung des Applikationsspezifischen Benutzermodus (in AP 1.4.3). Der generische Benutzermodus selbst wird danach als relativ unabhängiges Modul implementiert.
	AP-Nicht-Inhalte • Die Implementierung des applikations-spezifischen Benutzermodus erfolgt in 1.4.3 • Alle Hardware-Relevanten Teile werden suspendiert. Sie werden erst in 1.5.2 als zusätzliches Modul integriert.

	AP-Ergebnisse • E1: Alle für beide Benutzermodus relevanten Funktionalitäten implementiert • E2: Alle für den generischen Benutzermodus festgelegten Funktionalitäten implementiert
	AP-Leistungsfortschrittsmessung • Sobald E1 vorliegt: 30% • Sobald E2 zu 80% vorliegt (80% der festgelegten Funktionalitäten impl.): 70% • Sobald E2 vollständig vorliegt: 100%
1.5.1 BCI-SW-Modul an Schnittstelle anbinden	AP-Inhalt: • Das vom Hersteller zur Verfügung gestellte Simulink-Modul an die in 1.3.1 entwickelte Schnittstelle anbinden • Es ist zu beachten, dass dieses AP vollständig abgeschlossen sein muss, um sinnvoll die AP 1.5.2 und 1.5.3 zu bearbeiten. • Es wird dabei zwangsläufig zu Testläufen mit der Hardware kommen; es ist darauf zu achten, dass diese Testläufe einige Tage vorher bekannt und mit dem Institut, bei dem die Hardware zur Verfügung steht, abgestimmt wird.
	AP-Nicht-Inhalte • Alle Entwicklungsarbeiten, die die FF-Extension selbst betreffen, erfolgen erst in 1.5.2. Tests, die die Datenbübertragung zwischen Simulink und Firefox betreffen, sollen die in 1.3.2 entwickelte Schnittstellen-Demo-Extension verwenden.
	AP-Ergebnisse • E1: Der um den HW-relevanten Teil erweiterte Schnitstellen-Code auf Simulink-Seite steht zur Verfügung • E2: Der resultierende Code wurde mit der in 1.3.2 entwickelten Schnittestellen-Demo-Extension getestet.
	AP-Leistungsfortschrittsmessung • Wenn E1 vorliegt: 75% • Wenn E2 vorliegt: 100%

1.5 Projektbalkenplan



	Name	Dauer
1	**Projektmanagement**	77 tage?
2	Projekt gestartet	0 tage
3	Projektstart	2 tage
4	Projektkoordination	74 tage
5	Projektabschluss	2 tage
6	Projekt abgeschlossen	0 tage?
7	**Recherche und Know-How**	24 tage
8	BCI-Papers suchen	3 tage
9	FF-Extension-Development Know-How aneignen	10 tage
10	Matlab und Simulink Basics aneignen	10 tage
11	Simulink-BCI-Modul: Verarbeitungsschritte verstehen	5 tage
12	Schnittstellen zwischen Simulink und FF finden	15 tage
13	Bakk-Arbeit: Doku der Rechercheergebnisse	5 tage
14	Fundierte Entscheidung für die Impl. einer best. Schnittstelle gefallen	0 tage
15	**Schnittstelle entwickeln**	7 tage
16	Schnittstelle zwischen Simulink und FF implementieren	5 tage
17	Demo-Extension implementieren	5 tage
18	Bakk-Arbeit: Doku der Schnittstelle	3 tage
19	Simple FF-Extension demonstriert geeignete Datenübetragung	0 tage
20	**Extension entwickeln**	19 tage?
21	Hardware-Emulation per Tastendruck	2 tage
22	generischen Benutzermodus implementieren	14 tage
23	Applikationsspezifischen Benutzermodus implementieren	14 tage
24	Funktionalität der Extension testen	3 tage
25	Demo-Web-Applikation für Nutzung der Extension implementieren	5 tage
26	Bakk-Arbeit: Doku der Extension	9 tage?
27	Funktionalität der Extension aus Sicht der Web-Applikation hergestellt	0 tage
28	**Hardware anbinden**	9 tage?
29	BCI-SW-Modul an Schnittstelle anbinden	3 tage
30	Hardware-spezifischen Teil der Extension implementieren	7 tage?



Im obigen Balkenplan sind die Arbeitspakete der einzelnen Phasen (rote und blaue Balken), die Phasen selbst (schwarze Linien) und die Meilensteine (schwarze Rauten) eingetragen. In vielen Phasen werden Arbeitspakete parallel abgearbeitet, zumal dadurch einerseits die Integration der AP-Ergebnisse leichter fällt (Projektkoordination) und andererseits die Leistung des 2-Mann-Teams effizient ausgenutzt werden kann.

Beispielsweise startet das AP „Applikationsspezifischen Benutzermodus implementieren“ einige Tage nach Start des Arbeitspakets „generischen Benutzermodus implementieren“ und läuft dann einige Tage parallel. Der Grund besteht darin, dass zunächst beide Teammitglieder ihre Zeit und Energie für die Entwicklung des generischen Benutzermodus aufwenden sollen. Später, wenn die Grundzüge klar sind, beginnt eines der Teammitglieder, den Source-Code – unter Wiederverwendung wichtiger Teile des generischen Benutzermodus – in Richtung des applikationsspezifischen Benutzermodus zu entwickeln.

Parallel dazu testen beide Teammitglieder ihren Code. Dabei schreibt einer davon auch eine kleine Web-Applikation, der vor allem die Funktion des applikationsspezifischen Benutzermodus testen und demonstrieren soll.

Wichtige Ergebnisse werden bei all diesen Tätigkeiten sofort schriftlich festgehalten und dokumentiert. Der Grund dafür ist die Annahme, dass wichtige Zusammenhänge zum Zeitpunkt der Entwicklung ohne größeren Aufwand dargestellt werden können, während eine zeitversetzte Dokumentation zusätzlichen Aufwand oder eine niedrigere Qualität bedeuten kann.