ChameleonLab: Ein Ansatz für integriertes Wissens- und Workflow-Management in biomedizinischen Forschungslaboratorien unter Verwendung kryofunktionaler Speicherchips

Dissertation
zur Erlangung des Grades
des Doktors der Ingenieurwissenschaften
der Naturwissenschaftlich-Technischen Fakultät III
Chemie, Pharmazie, Bio- und Werkstoffwissenschaften
der Universität des Saarlandes

von
Dipl.- Inform. Christopher H.P. Durst

Saarbrücken, 2009

Tag des Kolloquiums:	04. Mai 2010
Dekan:	Prof. Dr. Stefan Diebels
Berichterstatter:	Prof. Dr. Günter R. Fuhr
	Prof. Dr. Dr. h. c. mult. Günter Hotz
Vorsitz:	Prof. Dr. Heiko Zimmermann
Akad. Mitarbeiter:	Dr. Tarek Hakki

Gekürzte Zusammenfassung

Diese Dissertation beschreibt Konzeption und Entwicklung eines Systems zum Wissens- und Workflow-Management in biomedizinischen Forschungslaboratorien als Lösungsansatz für Probleme bei konventioneller Laborarbeit. Forschungskontext ist die Entwicklung einer Technologieplattform für Biobanken, deren Kernelement eine Kopplung von Probe und elektronischem Speicherchip ist. Ursprünglich zur Verbesserung beim Management deklarativen Probenwissens gedacht, wird die zusätzliche Nutzung der Speicherchips zur Bewahrung, Verteilung und Nutzung prozeduralen Wissens konzipiert, um die Qualität von Proben bei ihrer Bearbeitung und die Qualität von Probenwissen bereits während seiner Generierung und Akquise sicherzustellen. Beginnend bei einer Anforderungsanalyse wird ein erstes Testsystem zur schrittweisen Benutzerführung unter Abarbeitung elektronischer Workflow-Definitionen konzipiert und umgesetzt. Darauf aufbauend werden in Untersuchungen, Evaluierungen und Benutzerbefragungen weitere Anforderungen identifiziert, die unter Einbeziehung von Aspekten kollaborativer Forschung zur Konzeption und Entwicklung eines CAQ-System verwendet werden, das dezentrales Wissens- und Workflow-Management integriert, benötigtes prozedurales Wissen zur Verfügung stellt, generiertes Wissen akquiriert und somit individuelle Einflüsse minimieren und Reproduzierbarkeit und Vergleichbarkeit etablieren kann. Zusätzlich werden Datenstrukturen konzipiert, die zur Integration von Probenröhrchen mit dezentralen Speichern in eine elektronische Lagertank-Infrastruktur benötigt werden.

Abstract

This PhD thesis deals with the conceptual design and development of a knowledge and workflow management system for biomedical research laboratories as an approach to solve problems associated with the conventional way of doing laboratory work. Research context is the development of a cutting-edge technology platform for biobanks. Its gist is the physical coupling of sample and electronic memory chip. Primarily focusing on improvements in managing sample data, the additional utilisation of the chips for preserving, distributing and using procedural knowledge is designed to assure sample quality during preparation and to assure knowledge quality already during its creation and acquisition. Starting from a requirements analysis, a first test system aiming at user guidance by procssing electronic workflow definitions is designed and developed. Based on this, further examinations and user evaluations show more requirements. Considering also aspects and demands coming from collaborative research scenarios, a CAQ system is conceptually designed and developed which integrates decentralized knowledge and workflow management, provides detailed procedural knowledge to reduce the impact of individual tacit knowing, acquires generated sample knowledge, and thus can establish reproducibility and comparability. Additionally, data structures are designed which are required to interface sample tubes equipped with memory chips with an electronic cryopreservation infrastructure.

1 Einleitung

Vitale Zellen werden in Humanmedizin und anderen biomedizinischen Bereichen für vielfältige Zwecke eingesetzt. Besonders hervorzuheben sind ihre etablierten diagnostischen wie auch therapeutischen Anwendungen, beispielsweise in Transfusions- und Transplantationsmedizin[1], aber auch ihre Bedeutung hinsichtlich wissenschaftlicher Forschung und dem Transfer resultierenden Wissens. Dieser Transfer erweitert kontinuierlich das Spektrum medizinischer Verfahren, zum Beispiel um Ansätze für die Gewinnung pluripotenter[2] Stammzellen aus adulten Organismen[3], und ermöglicht unter anderem die Entwicklung neuer Medikamente[4], Immunsera[5] und Impfstoffe[6]. Die erfolgreiche Eindämmung einer Vielzahl lethaler Epidemien läßt sich zum großen Teil auf die koordinierte Durchführung geeigneter Schutzimpfungen wie auch auf die Behandlung von Erkrankungen durch geeignete Medikamente zurückführen[7].

Die verschiedenen Einsatzmöglichkeiten unterschiedlichster Zelltypen erfordern die Entwicklung und iterative Optimierung spezialisierter und teils sehr komplexer Methoden, die in biomedizinischen Forschungslaboratorien zum Bearbeiten, Vorbereiten oder Verwenden vitaler Zellen benötigt werden. Solche Methoden werden in den Biowissenschaften als sogenannte *Protokolle*[8] formuliert[9]. Protokolle beinhalten hauptsächlich prozedurales Wissen in

---

[1] Eine solche therapeutische Anwendung ist die Transplantation von *CD34*-Stammzellen aus Nabelschnurblut, zusammenfassend beschrieben in [Bradley 2005].

[2] Als *pluripotent* werden Stammzellen bezeichnet, die in Zellen aller drei Keimblätter (Entoderm, Ektoderm und Mesoderm) sowie in Zellen der Keimbahn ausdifferenzieren können.

[3] beschrieben in [Kruse 2004]

[4] Beispielhaft seien hier erwähnt: (1) die Entdeckung der antibakteriellen Wirkung des Penicillins [Fleming 1929] und (2) der Transfer dieses Wissens zur Humantherapie [Chain 1940].

[5] Immunsera zur Postexpositionsprophylaxe und zur passiven Impfung enthalten aufgereinigte spezifische Antikörper.

[6] Impfstoffe enthalten Antigene zum induzierten Exprimieren spezifischer Antikörper und Immunzellen. Sie werden eingesetzt, um spezifische Immunität gegenüber bestimmten pathogenen Mikroorganismen zu erreichen.

[7] Ein Beispiel dafür ist die Reduzierung der Sterblichkeit bei Masern, beschrieben in [CDC 2007].

[8] Die Semantik des Begriffs *Protokoll* weicht in der Biologie völlig von dem in technischen Disziplinen üblichen Verständnis (‚Steuerungsverfahren oder Betriebsvorschriften für die Kommunikation zwischen Geräten auf einem Datenbus') ab. Biologen bezeichnen jede Anleitung zur Probenhandhabung oder –manipulation als Protokoll. Hinsichtlich ihrer undefinierten Detailschärfe sind Protokolle mit Kochrezepten vergleichbar.

Form der erforderlichen Prozeßschritte, die bei Präparation oder Anwendung spezifischer Proben auszuführen sind, und geben die Abfolge der Schritte vor. Im Bereich der Diagnostik beispielsweise fokussieren Protokolle auf die erforderlichen Arbeitsschritte zum Nachweisen bestimmter pathogener Mikroorganismen, signifikanter Antikörper oder sonstiger Marker[10].

Auch das langfristige Vorhalten vitaler Zellen in Biobanken[11] unter Bewahrung ihrer Funktionalität mit dem Ziel späterer Verwendung erfordert die Anwendung geeigneter Protokolle[12]. Eine Langzeitlagerung wird grundsätzlich durch den potentiellen Nutzen motiviert, der hinsichtlich retrospektiver Diagnostik, therapeutischer Anwendungen und wissenschaftlichem Erkenntnisgewinn erwartet wird. Die gegenwärtig einzige wissenschaftlich akzeptierte Methode für eine solche Langzeitlagerung ist die sogenannte *Kryokonservierung*[13]. Sie beruht auf einer geeigneten Vorpräparation vitaler Zellen, dem anschließenden kontrollierten Einfrieren gemäß zelltyp-optimierter Temperaturprofile und der anschließenden Lagerung bei Temperaturen unterhalb von -130°C. Solche Lagerbedingungen werden mit Hilfe von tiefkaltem, flüssigem Stickstoff (Siedepunkt –196°C) realisiert. Zellen werden entweder in einer Gasphase oberhalb flüssigen Stickstoffs oder in flüssigem Stickstoff gelagert[14]. Bei der letztgenannten Form der Kryokonservierung besteht jedoch die Gefahr einer wechselseitigen Infizierung der Proben (*Kreuzkontamination)*[15].

In der Literatur werden aus informatischer Sicht im Zusammenhang mit der Langzeitlagerung vitaler Zellen mehrere Schwierigkeiten[16] genannt, zum einen hinsichtlich langfristiger Verfügbarkeit und Sicherheit probenbezogenen Wissens[17] und zum anderen hinsichtlich der zuverlässigen Zuordnung dieses Wissens zu den jeweiligen Proben. Eine weitere wesentliche Schwierigkeit bei der Langzeitlagerung von Proben sind sich ändernde Informa-

---

[9] Die Formulierung als Protokoll entspricht der Externalisierung impliziten oder der Kombination expliziten Wissens gemäß dem SECI-Modell nach [Nonaka 2000], das Formen des Wissensaustauschs beschreibt.

[10] Die Expression bestimmter Marker in der Zellmembran ist symptomatisch für bestimmte Erkrankungen, kann aber auch Ansatzpunkt für ihre therapeutische Behandlung sein, wie beispielsweise der für bösartiges Wachstum humaner Prostata-Zellen signifikante Marker TRPV6, beschrieben in [Fixemer 2003].

[11] teilweise bis zu einer Dauer von mehreren Jahrzehnten

[12] Siehe auch [Tijssen 2008]

[13] Der Begriff ist vom griechischen Wort *kryos* (Eis, Kälte) abgeleitet. Wegweisende Forschungsergebnisse, speziell auch hinsichtlich der Verwendung von Kryoprotektiva, sind unter anderem beschrieben in [Polge 1949].

[14] In flüssigem Stickstoff entspricht die Lagertemperatur von Proben maximal der Siedetemperatur. Bei der Lagerung in Gasphase ist ein Temperaturgradient innerhalb des Lagertanks zu berücksichtigen.

[15] Eine Untersuchung solcher Risiken wird in [Bielanski 2000] beschrieben.

[16] Schwierigkeiten aus informatischer Sicht werden unter anderem in [Ölund 2007] beschrieben.

[17] Hier sind sowohl prozedurales Wissen in Form von Protokollen wie auch deklaratives, die Proben und ihre Eigenschaften beschreibendes Wissen gemeint.

tionsanforderungen[18]. Diese können aus biomedizinischem Fortschritt resultieren oder sich aus Änderungen im Anwendungsspektrum der jeweiligen Biobank ergeben. Künftige informationstechnische Anforderungen sind nicht von vornherein absehbar und erfordern daher eine weitgehende Flexibilität von Datenstrukturen und Datenbanken zur Wissensbewahrung[19]. Probenbezogenes Wissen liegt in der Regel in proprietär strukturierten Datenbanken vor, deren Datenaustauschformate nicht zwangsläufig miteinander kompatibel sind, so dass ein Wissensaustausch zwischen mehreren Biobanken oder anderen Institutionen mit weiteren Problemen verbunden ist. Als weitere Schwierigkeit ist die oftmals unterschiedliche Qualität und damit fehlende Vergleichbarkeit von deklarativem Probenwissen zu nennen, die aus unterschiedlichen Begriffsontologien, aber auch aus unterschiedlicher Qualität der Ausführung von Protokollen in verschiedenen Biobanken oder Laboratorien resultieren kann.

Im Allgemeinen ist jedes Protokoll auf jeweils einen Zelltyp und für eine spezielle Verwendung dieses Zelltyps optimiert. Daher kann eine irrtümliche Präparation einer Probe mit einem nicht geeigneten oder veralteten Protokoll zum Entstehen falschen deklarativen Wissens führen[20] und somit die Wissensbasis einer Biobank korrumpieren. Eine Fehlpräparation oder Verwechselung von Zellen kann im Zusammenhang mit der Herstellung zellbasierter Medikamente für humantherapeutische Anwendung aber auch zu besonderen Gefährdungen[21] von Patienten führen. Die zuverlässige Zuordnung von Proben und abzuarbeitenden Protokollen ist daher in biomedizinischen Laboratorien und Biobanken neben der zuverlässigen Zuordnung deklarativen Probenwissens von immenser Wichtigkeit. Dennoch hat sich in der Vergangenheit lediglich eine referenzierende Zuordnungsmethodik etabliert: eine Kennzeichnung[22] am Probengefäß stellt die logische Verknüpfung zu dem eigentlichen Probenwissen dar, das räumlich von der Probe getrennt in unterschiedlichsten Datenbanken[23] oder Karteikartensystemen organisiert ist. Aus dieser Zuordnungsmethodik ergibt sich eine reale Gefahr von Verwechselungen und Fehlzuordnungen, beispielsweise durch Verlust oder Beschädigung von Probenbeschriftungen. Lange Lagerzeiten und der Austausch von Proben

---

[18] nachgewiesen in [Durst 2003]

[19] Problematisch ist in diesem Zusammenhang das Entstehen verschiedener Versionen von Datenstrukturen und neuer Datenformate während der Laufzeit von Biobanken, denn ein Zugriff auf das Wissen unterschiedlich alter Proben muß zu jedem Zeitpunkt möglich sein.

[20] beispielsweise in Form einer falschen Diagnose

[21] Ein Beipiel hierfür ist die Abstoßungsreaktion gegenüber HLA-inkompatiblem Knochenmark.

[22] Die bisher etablierten Kennzeichnungen reichen von handschriftlichen Probennummern auf Probenröhrchen über die Verwendung von Etiketten mit Klartext oder Barcode bis hin zur Verwendung von RFID-Tags [Bettendorf 2005].

[23] Datenaustausch und Data-Mining unter Einbeziehung unterschiedlicher Datenbankstrukturen von Biobanken werden unter anderem in [Ölund 2007] beschrieben.

erhöhen diese Risiken. Verschiedene Versionen von Protokollen unterliegen durch referenzierende Zuordnungsmethodik ebenfalls einer realen Verwechselungsgefahr.

Für die Bearbeitung von Proben werden in biomedizinischen Forschungslaboratorien im Allgemeinen Papierausdrucke[24] zugeordneter Protokolle genutzt. Je nach Zelltyp und Verwendung von Proben gelten bei der Ausführung von Protokollen unterschiedliche Regularien zur Dokumentationsakquise. In biomedizinischen Forschungslaboratorien erfolgt die Labordokumentation meist handschriftlich in Laborbüchern. Eine sekundengenaue Aufzeichnung von Ereignissen erfolgt in biomedizinischen Forschungslaboratorien nicht. Als problematisch bei der Dokumentationsakquise anzusehen sind die von Person zu Person unterschiedliche Detailschärfe, Sorgfalt und individuelle Prioritäten, die unmittelbare Auswirkungen auf Qualität und Vergleichbarkeit des deklarativen Probenwissens haben[25]. Bestenfalls wird handschriftlich notierte Labordokumentation zu einem späteren Zeitpunkt noch elektronisch erfasst und in eine Datenbank eingepflegt[26]. Auch Labordokumentation wird der bearbeiteten Probe unter Verwendung referenzierender Verfahren zugeordnet. Ihre obligatorische Archivierungsdauer ist von Zelltyp und Anwendung abhängig[27].

Nicht nur die Qualität der Dokumentationsakquise ist neben der erforderlichen zuverlässigen Zuordnung von Protokollen und Dokumentation von höchster Wichtigkeit, sondern auch die Qualität der Laborarbeit selbst. Die Einhaltung bestimmter Normen[28], die grundsätzliche Arbeitsweisen regeln, ist für bestimmte biomedizinische Bereiche verpflichtende Voraussetzung. Unabhängig von solchen organisatorischen Reglementierungen hängt das Ergebnis jeder einzelnen Präparation unmittelbar von der konkreten Ausführung des entsprechenden Protokolls ab. Voneinander abweichende Einwirkzeiten bestimmter Reagenzien können beispielsweise gravierende Auswirkungen auf Zellproben haben, etwa auf die Vitalitätsrate bei der Kryokonservierung[29] oder auf die Ergebnisse lichtmikroskopischer

---

[24] Je nach der geltenden Sicherheitsstufe eines Labors ist die Verwendung von Papier auch wegen des Risikos der Kontamination problematisch. Dies gilt insbesondere auch für handschriftliche Dokumentation.

[25] Zusätzlich sind dadurch eine exakte Reproduzierbarkeit von Ergebnissen und eine genaue Korrelation biologischer Effekte mit zeitlichen Aspekten nicht möglich.

[26] Die Transkription handschriftlicher Labordokumentation in ein elektronisches System ist eine weitere potentielle Fehlerquelle (Unleserlichkeit, Fehlzuordnungen, mangelnde Sorgfalt).

[27] Für den Bereich der Herstellung zellbasierter Arzneimittel beispielsweise gelten die Vorschriften des Arzneimittelgesetztes und spezifische weitergehende Richtlinien.

[28] Beispiele: *Good Manufacturing Practice* (GMP) und *Good Clinical Laboratory Practice* (GCLP)

[29] Bestimmte für die Kryokonservierung verwendete Reagenzien haben bei Raumtemperatur zytotoxische Wirkung, beispielsweise das Kryoprotektanz *Dimethylsulfoxid* (DMSO).

Vitalitätstests[30]. Auch sehr spezifische Zellfunktionalitäten werden durch die jeweilige Protokollausführung beeinflusst. So variiert beispielsweise die Fähigkeit mononukleärer Zellen aus peripherem Blut (*PBMCs*) zur zellulären Immunantwort in Abhängigkeit kleinster Unterschiede bei der Kryokonservierung oder beim Waschen [Disis 2006][31]. Damit ist die Qualität der Erzeugnisse und des entstehenden deklarativen Wissens unmittelbar von der Qualität der Probenbearbeitung abhängig. In diesem Zusammenhang ist die bereits anfangs erwähnte undefinierte Detailschärfe von Protokollen ein bedeutsames Problem. Während manche Protokolle eine sehr detaillierte Vorgehensweise für jeden Prozeßschritt vorgeben, sind andere weniger detailliert und erfordern die Substitution fehlender Details durch individuelles *implizites Wissen*[32]. Die Ausführung von Protokollen unterliegt somit den unterschiedlichen individuellen Einflüssen der ausführenden Personen. Dies kann zu gravierenden Unterschieden zwischen den Präparationsergebnissen führen, die im Nachhinein kaum auf Nuancen bei der Protokollausführung zurückgeführt werden können, denn die Qualität der Dokumentationsakquise variiert wie oben beschrieben ebenfalls. Die entstehende Labordokumentation ermöglicht somit weder eine verläßliche Vergleichbarkeit von Präparationsergebnissen oder deklarativem Probenwissen noch eine verläßliche Reproduzierbarkeit von Präparationen[33].

Besonders gravierende Auswirkungen haben die geschilderten Schwierigkeiten in kollaborativen Forschungsszenarien, deren Intention der kooperative Aufbau einer gemeinsamen Wissensbasis ist. Ein Beispiel für ein solches Forschungsszenario ist die *Collaboration for*

---

[30] Bestimmte Farbstoffe (Beispiel: *Ethidiumbromid*) haben zellschädigende Wirkung und können bei zu langer Einwirkzeit Vitalitätstests negativ beeinflussen.

[31] Quantitative Unterschiede zwischen frischen und kryokonservierten PBMCs hinsichtlich spezifischer Funktionalitäten werden in [Weinberg 2000] beschrieben.

[32] Der Begriff des impliziten Wissens [Polanyi 1985] bezeichnet individuelles Know-How, das im Wesentlichen auf Erfahrungen und Routinisierung ehemals bewußter Handlungen beruht. Implizites Wissen steht in direktem Zusammenhang mit erfahrungsgeleitetem Arbeitshandeln und hat direkten Einfluß auf Art und Güte von Handlungen.

[33] Bislang wurden für spezialisierte Laboratorien mit hohem Probenaufkommen verschiedene Automatisierungsansätze etabliert, um die Qualität von Probenpräparationen zu vereinheitlichen. Diese Systeme sind jeweils auf eine überschaubare Anzahl verschiedener Protokolle mit jeweils hohem Probenaufkommen optimiert, etwa im *High-Throughput-Screening*. Die automatische Akquise dabei anfallender Labordokumentation erfolgt durch *Laborinformations-Management-Systeme* (LIMS). In biomedizinischen Forschungslaboratorien sind solche Systeme jedoch aufgrund der Vielzahl verschiedener Protokolle, die kontinuierlich iterativer Optimierung unterliegen, nicht sinnvoll anwendbar. Zudem würden vollautomatisierte Prozeßstrecken die Menge an kompatiblen Protokollen stark einschränken.

AIDS Vaccine Discovery (*CAVD*)[34], ein internationales Forschungsnetzwerk zur Entwicklung eines HIV-Impfstoffes. In der Vergangenheit war die Entdeckung eines solchen Impfstoffes unter anderem an der sehr hohen Mutationsrate des HI-Virus und der daraus resultierenden Vielzahl an HIV-Varianten gescheitert[35]. Die kollaborative Forschung der CAVD setzt auf den Aufbau einer gemeinsamen Wissensbasis und führt komparative Impfstoffstudien durch. Gegenstand dieser Impfstoff-Studien sind unter anderem Untersuchung und Vergleich zellulärer Immunantworten HIV-infizierter Zellproben auf potentielle Impfstoffe. Dazu unbedingt erforderlich ist die qualitative und quantitative Vergleichbarkeit der gemessenen Immunantworten. Dies setzt eine vergleichbare, einheitliche und nachvollziehbare Ausführung der benötigten Laborprotokolle voraus, da selbst kleinste Abweichungen relevante Auswirkungen auf die Funktionalität der relevanten Zellen haben können. Gravierende Auswirkungen auf diese Zellen haben auch Transportvorgänge[36].

Das Fraunhofer-IBMT entwickelt innerhalb der CAVD das *Global HIV Specimen Research Cryorepository* (*GHRC*) für HIV-infizierte Blutproben, Virusisolate und Reagenzien. Weitere Aufgaben des GHRC-Konsortiums[37] sind die Entwicklung innovativer Lagertechnologien für Proben, die Optimierung von Laborprotokollen und die Entwicklung von Laborstandards. Insbesondere werden in diesem Zusammenhang Methoden für die zuverlässige Zuordnung probenrelevanten Wissens[38] benötigt. Für deklaratives Probenwissen wurde in [Durst 2003] bereits ein Konzept zu dezentralem Wissensmanagement[39] motiviert, das eine Kopplung von Probe und kryotolerantem elektronischem Speichermedien vorsieht. Ein solcher Ansatz kommt Forderungen[40] nach Transparenz in Biobanken entgegen. Ausgehend von ersten experimentellen Evaluierungen zur Verwendbarkeit von Flash-Speicher-

---

[34] CAVD umfaßt vierzehn Impfstoffkonsortien und fünf zentrale Service-Einrichtungen. Es sind 89 Institutionen aus 22 Ländern involviert.

[35] Im September 2007 wurde eine klinische HIV-Impfstoffstudie von Merck abgebrochen, da die Anfälligkeit für eine HIV-Infektion bei geimpften Personen unter bestimmten Umständen sogar höher zu sein schien als bei der ungeimpften Vergleichgsgruppe [Sekaly 2008].

[36] Auswirkungen von Transportvorgängen zwischen verschiedenen Laboratorien auf die Funktionalität von T-Lymphozyten und die daraus entstehende Ergebnisverfälschung gemeinsamer Studien wird beschrieben in [Betensky 2000].

[37] Das GHRC-Konsortium umfaßt neben dem Fraunhofer-IBMT folgende Institutionen: die World Health Organisation (WHO) in Genf, das National Institute for Biological Standards and Control (NIBSC) in London, das San Raffaele Scientific Institute (DIBIT) in Mailand, die University of Lund, die Universität des Saarlandes, die University of Washington und das National Institute of Health (NIH).

[38] Im Bereich kollaborativer Impfstoffforschung können Fehlzuordnungen die gemeinsame Wissensbasis verfälschen.

[39] Die Aufgaben von Wissensmanagement sind unter anderem erläutert in [Scheer 2002].

[40] Gefordert in [Pearson 2004].

medien unter kryogenen Bedingungen in [Ihmig 2003] und [Ihmig 2004] und parallel zur Entwicklung einer entsprechenden Technologieplattform war ein Ziel der vorliegenden Arbeit die Konzeption von Datenstrukturen für dezentrale Wissensbewahrung. Ein weiteres Ziel war die Entwicklung von Konzepten für die zuverlässige Zuordnung von Laborprotokollen und Labordokumentation zu Proben. Weitere Ziele waren die Konzeption und prototypische Implementierung eines Informationssystems für biomedizinische Forschungslaboratorien, das eine reproduzierbare Ausführung von Laborprotokollen und die Akquise vergleichbarer Labordokumentation ermöglicht und die zuvor erarbeiteten Konzepte zur zuverlässigen Wissenszuordnung und –bewahrung einbezieht. Der wissenschaftliche Kontext und die Aufgabenstellung der vorliegenden Arbeit sind somit stark interdisziplinär geprägt und lassen sich mit Hilfe folgender Abbildung skizzieren:

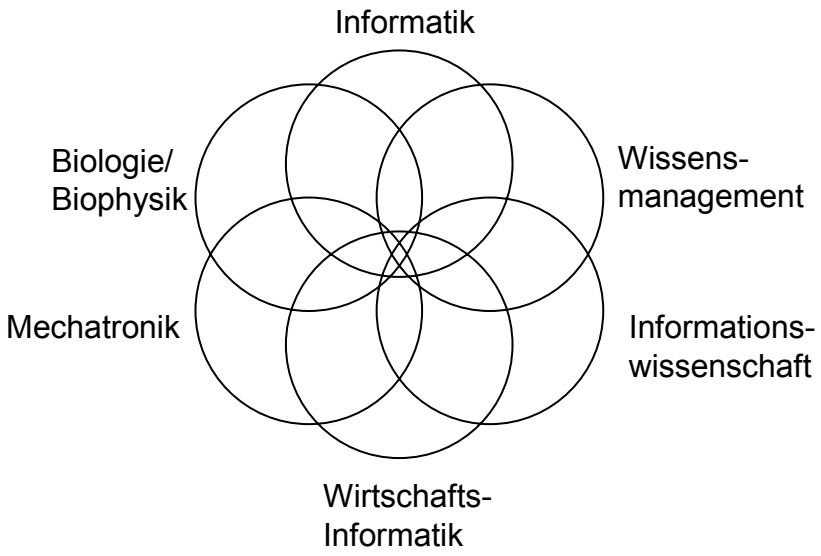

Abbildung 1: Interdisziplinarität des wissenschaftlichen Kontextes und des Arbeitsgebietes der vorliegenden Dissertation. Biologische Anforderungen (auch im Zusammenhang mit biophysikalischen Besonderheiten bei der Kryokonservierung von Proben) werden vor dem Hintergrund mechatronischer Entwicklungen informatisch umgesetzt. Eine zentrale Rolle nehmen dabei Wissensmanagement und Wirtschaftsinformatik ein. Die Repräsentation von Wissen involviert zusätzlich informationswissenschaftliche Aspekte.

2 Grundlagen

2.1 Wissen im Kontext von Biobanking

Mit biologischen Proben ist im Allgemeinen umfangreiches Wissen assoziiert, das den wissenschaftlichen Wert von Proben und Probensammlungen ausmacht. Die Aufgaben von Biobanken bestehen somit nicht nur im Lagern der Proben, sondern auch im Management des zugehörigen Wissens. Das vorliegende Kapitel behandelt Aspekte aus Wissenstheorie und Wissensmanagement, die im weiteren Verlauf der vorliegenden Arbeit von Bedeutung sind.

Grundsätzlich kann probenbezogenes Wissen grob unterteilt werden in:

(1) Wissen, das unabhängig von einer Probenbearbeitung zur Verfügung steht oder erhoben werden kann (Spenderdaten, Probenherkunft, Anamnese, Datum und Zeit der Entnahme, Probenart, usw.)

(2) Wissen, das zur Lagerung / Bearbeitung / Verwendung der Probe erforderlich ist (Einfrierkurven, Bearbeitungsvorschriften, usw.)

(3) Wissen, das durch Lagerung, Bearbeitung oder Probenverwendung entsteht (Lagerdaten, Labordokumentation, Laborergebnisse, Diagnosedaten, festgestellte Eigenschaften, usw.)

2.1.1 Wissenstypen

Ryle [Ryle 1949] unterscheidet die beiden Wissenstypen *know-how* als Wissen über Vorgehensweisen und Strategien (auf dem gemäß [Reinmann-Rothmeier 1998] Fertigkeiten beruhen) und *know-that* als Wissen über Fakten und Regeln. Gemäß [Herbig 2001] entspricht diese Unterscheidung im Wesentlichen der in der Literatur häufig verwendeten Unterscheidung zwischen *prozeduralem* und *deklarativem* Wissen [Oberauer 1993]. Entsprechend dieser Definitionen kann das oben identifizierte probenbezogene Wissen klassifiziert werden: (1) ist deklaratives Wissen, (2) ist prozedurales Wissen und (3) ist deklaratives Wissen, das als Resultat aus der Anwendung prozeduralen Wissens entsteht[41].

[41] Auf Korrektheit und Qualität des dabei entstehenden deklarativen Wissens wird weiter unten in diesem Kapitel eingegangen.

2.1.2 Wissensmodi

Als *explizites Wissen* bezeichnet man (vgl. [Büssing 1999]) verbalisierbares, reflektierbares und instruktiv vermittelbares Wissen, das durch bewußtes, beabsichtigtes Lernen [Reber 1977] oder durch explizites Lehren im Sinne einer sprachlichen Übermittlung [Neuweg 1998] erworben wird. Dieser Vorgang kann ohne direkten Kontakt mit dem Gegenstandsbereich stattfinden. Explizites Wissen ist, da es immer bewußt ist, mittels Zeichen (beispielsweise Sprache oder Schrift) kodierbar und kann in Büchern oder Speichermedien aufbewahrt werden [Frischmuth 2002]. Es ist vom Wissensträger unabhängig und läßt sich kommunizieren, beispielsweise in Arbeitsanweisungen, wissenschaftlichen Publikationen oder dokumentierten Erfahrungsberichten.

Man geht davon aus, daß es neben dem expliziten Wissensmodus einen weiteren, schwerer zugänglichen Wissensmodus gibt, der im alltäglichen Leben automatisch und unbewußt bei der Ausübung von Tätigkeiten aktiviert wird, speziell in Bereichen, in denen eine Person umfangreiche gegenständliche Erfahrung besitzt. Als Beispiel für diesen Modus wird häufig die Muttersprache genannt (vgl. [Müller 2004]), die man hinsichtlich Betonung und Grammatik problemlos beherrsche, deren Regeln man aber beispielsweise einem Ausländer nur unzureichend erklären könne. Auch das Erkennen von Gegenständen anhand ihrer Merkmale, die nicht umfassend bewußt sind und daher nicht ausreichend erklärt werden können, ist ein Beispiel für diesen Wissensmodus. Ein weiteres Beispiel ist das Wissen über eine bestimmte Wegstrecke, die regelmäßig zurückgelegt wird, aber nur mit Mühe erklärt werden kann. Dieses Phänomen, „mehr zu wissen, als wir zu sagen wissen“ wurde erstmals in [Polanyi 1966] anhand menschlicher Wahrnehmung illustriert, und dieser Wissensmodus wird als *tacit knowing* bzw. *implizites Wissen* [Polanyi 1985] bezeichnet. Jedoch existiert keine einheitliche oder für alle Forschungsbereiche verbindliche Definition des Begriffs[42]. Im Gegensatz zu explizitem Wissen ist implizites Wissen aufgrund seiner mangelnden Bewußtheit zunächst nicht verbalisierbar und damit auch nicht instruktiv vermittelbar [Dienes 1997], allerdings beschreiben die *Possible-access-Position* und entsprechende experimentelle Untersuchungen [Reber 1989], daß implizites Wissen bewußt zugänglich sein kann bzw. nicht notwendigerweise unbewußt bleiben muß und somit verbalisiert und kommuniziert werden kann. Dennoch reflektieren die Aspekte der impliziten Datenbasis, die ohne weiteres explizierbar sind, nicht die gesamte Menge des impliziten Wissens einer Person bezüglich

[42] Über den Begriff des impliziten Wissens schreibt Neuweg (vgl. [Neuweg 2004]): „Der Begriff des impliziten Wissens vereinigt in sich nahezu alle Eigenschaften, die man sich von einem Terminus in der wissenschaftlichen Diskussion gerade nicht wünscht. Er ist, sich gleichsam selbst bestätigend, ausgesprochen unscharf, wird keineswegs einheitlich und im Rahmen verschiedener Theoriekontexte verwendet, die paradigmatisch partiell oder vollständig unverträglich sind, verbindet sich mit unterschiedlich starken Annahmen und besitzt Konnotationen, die ihn für Mystifizierungen anfällig machen.“.

einer Handlung[43]. Im Bereich biomedizinischer Labortätigkeit kann als eines vieler passender Beispiele für diesen Sachverhalt das Pipettieren genannt werden, das sensorisch-motorische Fertigkeiten, also komplexe sinnliche Wahrnehmung über mehrere Sinne und Zusammenhangswahrnehmung (‚Fingerspitzengefühl') involviert, die durch praktisches Ausüben der Tätigkeit ‚einverleibt' wurden, aber bei einer Explikation impliziten Wissens kaum kommuniziert werden können.

Gerade das Verwenden von Werkzeugen und die damit zusammenhängende sinnlich-körperliche Wahrnehmung wird von Polanyi im Zusammenhang mit impliziten Lernprozessen wie folgt beschrieben: „Unser Körper ist das grundlegende Instrument, über das wir sämtliche intellektuellen oder praktischen Kenntnisse von der äußeren Welt gewinnen. (...) Werkzeuge können als empfindungsbegabte Verlängerungen unseres Körpers angesehen werden. Der Umstand, daß wir uns äußeren Dingen zuwenden, indem wir uns unseres Körpers gewahr werden, legt es nahe, die Reichweite unserer Körperempfindungen auszuweiten (...). In diesem Sinne können wir sagen, daß wir uns die Dinge einverleiben (...) oder umgekehrt, daß wir unseren Körper soweit ausdehnen, bis er sie einschließt (...)" [Polanyi 1985].

Folgende tabellarische Übersicht, angelehnt an eine Darstellung in [Büssing 1999], zeigt einige Merkmale von explizitem Wissen im Kontrast zu implizitem Wissen:

Explizites Wissen	**Implizites Wissen**
<u>Verbalisierbarkeit</u> • verbalisierbar, da bewußt • instruktive Vermittelbarkeit	• allgemein schwer verbalisierbar, da meist unbewußt, aber ‚possible-access'-Annahme
<u>Bewusstheit</u> • immer bewußt • reflektiert	• meist unbewußt • unreflektiert
<u>Akquisition</u> • bewußtes (explizites) Lernen • kein Kontakt zum Gegenstandsbereich erforderlich	• Routinisierung ehemals bewußter, expliziter Prozesse • unbewußtes (implizites) Lernen • durch Vorbilder und Musterbeispiele
<u>Sinnlichkeit</u> • Wissen basiert meist auf Wahrnehmungen über einen Sinn (beispielsweise visuell beim Lesen)	• Wissen basiert auf komplexen sinnlich-motorischen Wahrnehmungen

Tabelle 1: wesentliche Merkmale expliziten und impliziten Wissens im Vergleich. Der Vergleich fokussiert auf die Aspekte Verbalisierbarkeit, Bewusstheit, Akquisition und Sinnlichkeit. Aufgrund seines unbewußten und oftmals durch Routinisierung erfolgten Erwerbs ist implizites Wissens im Gegensatz zu explizitem Wissen meist unbewußt, schwierig verbalisierbar und kaum instruktiv zu vermitteln.

[43] Im Zusammenhang mit Expertensystemen wird die unvollständige Explikation von Expertise bezüglich handlungsleitenden Wissens als eine der Schwierigkeiten genannt [Kirsner 1998].

Folgende Tabelle zeigt in Anlehnung an [Neuweg 1998] einige Beispiele für implizites Wissen im Kontrast zu seinen jeweiligen expliziten Gegenstücken:

	Explizit	**Implizit**
Begriffswissen	ausdrückliches Wissen um die identifizierenden Merkmale der Objekte, Ereignisse und Situationen	Fähigkeit, Objekte, Ereignisse und Situationen als gleichartig zu erkennen
Erwartungswissen	Wenn-dann-Relationen	Wissen, welche Ereignisse üblicherweise aufeinander folgen
Handlungswissen	Verfahrensbeschreibungen, Ziele und Regeln	Wissen darüber, wie man eine Tätigkeit ausführt oder welche Handlungen wann angemessen sind

Tabelle 2: Beispiele für explizites und implizites Wissen im Vergleich. Die Unbewußtheit impliziten Wissens artikuliert sich in Fähigkeiten, unbewußten Erwartungen intuitiven Handlungen, während explizites Wissen in Form von Regeln, Beschreibungen und Relationen bewusst ist.

Explizites Wissen ist nach [Neuweg 1998] grundsätzlich an die „impliziten Operationen einer Person“ gebunden insofern, als explizites Wissen in Form von Wörtern oder sonstigen Symbolen verstanden werden muß, um eine Bedeutung zu erlangen. Sprache als Symbolik erfordert Sprachverständnis in Form entsprechenden impliziten Wissens (vgl. [Mulzer 2006]). Im Zusammenhang mit biomedizinischen Fachtermini ist ein entsprechend implizites Wissen ebenfalls erforderlich, beispielsweise muß dem Symbol ‚Pipettieren’ ein implizites Handlungswissen entsprechen, damit eine Handlungsinstruktion zum Pipettieren sinnvoll ausgeführt werden kann.

Die oben erläuterten Wissensmodi[44] und ihre Zusammenhänge werden hauptsächlich in der Kognitionspsychologie, der pädagogischen Psychologie und der Arbeitspsychologie thematisiert und erforscht. Im Rahmen der vorliegenden Dissertationsarbeit sind sie besonders hinsichtlich der Bedeutung impliziten Wissens im Arbeitskontext biomedizinischer Probenbearbeitung und hinsichtlich geeigneter Wissensvermittlung relevant, denn implizites Wissen wird in besonderem Maße mit Erfahrungswissen (im Sinne von [Böhle 1995]) und erfahrungsgeleitetem Arbeitshandeln in Verbindung gebracht: je mehr implizites Erfahrungswissen bei der Bewältigung von Arbeitssituationen notwendig ist, umso mehr wird diese Arbeit zu einer erfahrungsgeleiteten Arbeit [Carus 1996]. Personen gehen nach [Büssing 2002] in einen erfahrungsgeleiteten Handlungsmodus über, wenn explizite Informationsverarbeitung nicht möglich ist, so daß in solchen Arbeitssituationen erfahrungsgeleitetes Arbeitshandeln stattfindet. Unterschiede im Handeln können nach [Speelman 1998] auf Unterschiede im impliziten

[44] Der Vollständigkeit halber sei hier noch das Phänomen des *trägen Wissens* als weiterer, zum impliziten Wissen komplementärer Wissensmodus erwähnt. Dabei handelt es sich um verbales Wissen, das sich nicht in entsprechender Urteils- und Handlungskompetenz niederschlägt [Neuweg 2004a].

Wissen zurückgeführt werden. Dieser Aspekt ist zentral im weiteren Verlauf dieser Dissertation[45], unter anderem hinsichtlich unterschiedlicher Ausführung ‚per se' identischer biomedizinischer Protokolle. Das Konzept des erfahrungsgeleiteten Handelns (vgl. [Herbig 2001a], [Witt 1996]) beschreibt zusätzlich den Aufbau impliziten Wissens durch Erfahrung in der praktischen Arbeitstätigkeit ohne bewußtes Lernen. Dies deckt sich mit der Vorstellung Polanyis über die Aneignung von implizitem Wissen und seine Weiterentwicklung durch konkrete gegenständliche Erfahrung und mit der Vorstellung von [Myers 1992] über implizites Wissen als Erfahrungswissen, das „untaught" ist und „increased by experience in the job"[46]. Grundsätzlich werden meist die folgenden beiden Arten der Akquisition impliziten Wissens unterschieden: (1) durch Routinisierung (oder Wissenskompilierung) ehemals expliziten Wissens (deklarativer wie auch prozeduraler Art, beispielsweise von Instruktionen) im Sinne von Andersons ACT-Modell [Anderson 1992], nach dessen Vorstellungen explizites Wissen soweit kompiliert und durch Anwenden verinnerlicht wird, bis prozedurales Wissen nicht mehr dem Bewußtsein zugänglich ist (vgl. auch *Verinnerlichung* bzw. *Prozeduralisierung* in [Berry 1987], [Büssing 1999]; bei sehr komplexen oder selten auftretenden Aufgaben ist dies jedoch nicht immer vollständig möglich) und (2) durch das oben erwähnte unbewußte, nicht beabsichtigte Lernen (vgl. auch [Seger 1994], [Berry 1984], [Berry 1987]).

Gemäß [Herbig 2001b] ist anzunehmen, daß im Laufe wachsender Berufserfahrung auch implizites Wissen und Heuristiken aufgebaut werden, die falsch sind, aber durch mangelnde Bewußtheit nie explizit hinterfragt werden. Gleiches gilt für die unbewußte Aneignung komplexen prozeduralen Wissens, das ohne zugrundeliegende Regeln erlernt wurde und zu fehlerhaftem oder unangemessenem erfahrungsgeleiteten Arbeitshandeln führen kann [Lewicki 1987]. Nach [Gelman 1994] kann es sogar bereits bei Erlernen expliziten Wissens zu fehlerhaften Inhalten kommen. Im Bereich des Biobankings oder der Probenbearbeitung können durch fehlerbehaftetes Wissen Fehlpräparationen oder sonstige Abweichungen von der eigentlich beabsichtigten Handlung entstehen, die je nach Art der Probe oder ihrer Anwendung zu teils gravierenden Konsequenzen führen können.

---

[45] Die Qualität deklarativen Wissens, das durch Ausführung prozeduralen Wissens entsteht, ist gemäß den obigen Ausführungen unmittelbar abhängig vom Maß des für die Ausführung erforderlichen impliziten Wissens und seiner objektiven Korrektheit. Je mehr implizites Wissen erforderlich ist (beispielsweise zur Bestimmung von Probeneigenschaften oder zur Ausführung bestimmter Diagnosen), desto größer können die Abweichungen zwischen deklarativem Wissen sein, das von unterschiedlichen Personen generiert wird.

[46] Polanyi [Polanyi 1985] führt als Beispiel für diesen Sachverhalt an, daß die Exzellenz eines hervorragenden Arztes nicht auf seine gewissenhaftere Lektüre von Lehrbüchern zurückzuführen sei, sondern auf seine Fertigkeiten als Diagnostiker – eine persönliche Fähigkeit, die durch praktische Erfahrung erworben worden sei.

2.1.3 Wissensaustausch

Ein Modell, das die verschiedenen Formen des Wissensaustauschs und der Umwandlung der oben beschriebenen Wissensmodi innerhalb und zwischen Personen beschreibt, ist das *SECI-Modell* von Nonaka und Takeuchi [Nonaka 1997], [Nonaka 2000]. Es basiert auf vier Formen der Wissensumwandlung, die im Idealfall zirkulär stattfinden (*Spirale des Wissens*) und zu einem kontinuierlichen Wissenszuwachs führen:

- *Sozialisation*: implizites Wissen von Einzelpersonen wird dem impliziten Wissen anderer Einzelpersonen oder Personengruppen hinzugefügt, ohne expliziert zu werden. Das implizite Wissen wird nicht bewußt kommuniziert, sondern durch Interaktion und gegenseitige Beobachtung von Personen übertragen. Typische Beispiele für Sozialisation sind Meister-Lehrling-Beziehungen, Mentoring oder Cognitive Apprenticeship. Eine anschauliche Beschreibung dieser Beispiele findet sich beispielsweise in [Müller 2004]. Sozialisation setzt eine sehr enge Zusammenarbeit zwischen den Personen voraus und beschränkt sich somit auf eine geringe Anzahl simultan beteiligter Personen. Durch Sozialisation gelerntes Wissen unterliegt der Gefahr, zumindest in Teilen vom impliziten Wissen anderer Personen abzuweichen, das in anderen Personenkonstellationen erworben wurde.
- *Externalisierung*: bezeichnet in diesem Modell den Prozess der Umwandlung impliziten Wissens einer Person in formalisierbares, bewußt zugängliches und verbalisierbares, also explizites Wissen. Dieses Wissen wird somit anderen Personen zugänglich und kann Teil der gemeinsamen Wissensbasis einer „Learning Community“ [Mandl 2000] werden, in der durch Austausch individuellen Wissens und gemeinsamen Wissens der Gemeinschaft neues Wissen geschaffen wird. Externalisierung ist im Zusammenhang mit dem im folgenden Abschnitt 2.1.4 vorgestellten Prozeßmodell des Wissensmanagements Voraussetzung für Wissensrepräsentation und Wissenskommunikation und aus Sicht von Nonaka und Takeuchi der Schlüssel zur Wissensschaffung in Organisationen.
- *Kombination*: bezeichnet die Wissensgenerierung durch Kombinieren expliziten, formalisierten Wissens mehrerer Personen. Dieses Kombinieren kann durch Zusammenfügen, Sortieren oder mündlichen Austausch expliziten Wissens stattfinden und zu neuen Ideen führen.
- *Internalisierung*: bezeichnet den Vorgang der Verinnerlichung neu entstandenen, gemeinsamen expliziten Wissens durch jedes Individuum, meist durch praktische Anwendung. Explizites Wissen wird also durch Internalisierung wieder zu implizitem Wissen der einzelnen Personen und fließt in das erfahrungsgeleitete Handeln ein.

Folgende Abbildung zeigt die vier Formen der Wissensumwandlung gemäß Nonaka und Takeuchi:

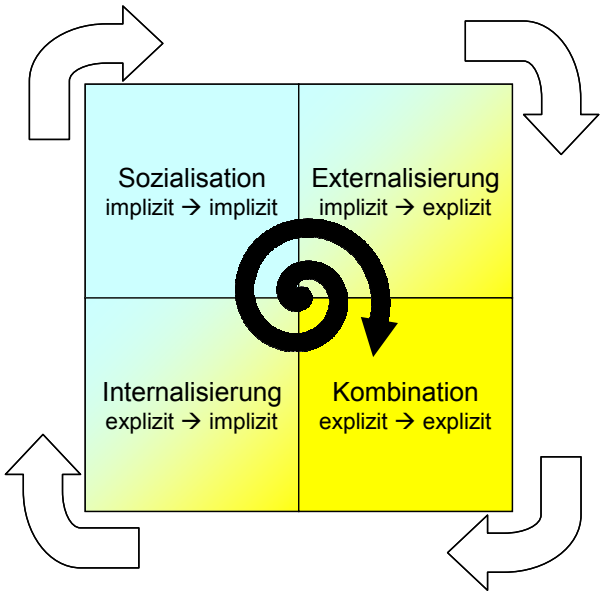

Abbildung 2: Die vier Formen der Wissensumwandlung gemäß Nonaka und Takeuchi, dargestellt als SECI-Modell mit der Spirale des Wissens in Anlehnung an eine Abbildung aus [Nonaka 2000]. Implizites Wissen wird durch hellblaue Farbe repräsentiert, explizites Wissen durch gelbe Farbe. Sozialisation: implizites Wissen wird von Einzelpersonen unbewußt dem impliziten Wissen einer Personengruppe hinzugefügt, ohne expliziert zu werden. Dies geschieht beispielsweise durch Beobachtung oder Interaktion bei sehr enger Zusammenarbeit. Externalisierung: implizites Wissen einer Person wird in bewusstes und verbalisierbares Wissen umgewandelt, das somit auch anderen Personen zugänglich wird und als Teil einer gemeinsamen Wissensbasis für den Prozeß der Kombination zur Verfügung steht. Kombination: aus formalisiertem und verbalisiertem Wissen wird neues explizites Wissen generiert. Internalisierung**:** explizites Wissens wird durch wiederholte Anwendung und Routinisierung zu unbewußtem, implizitem Wissen. Der iterative Wissenszuwachs einer individuellen oder gemeinsamen Wissensbasis, der aus mehrmaligem Durchlaufen der einzelnen Umwandlungsschritte resultiert, wird durch die abgebildete Spirale des Wissens repräsentiert.

2.1.4 Wissensmanagement

Wie eingangs erwähnt, ist eine der wesentlichen Aufgaben von Biobanken das Management von probenbezogenem Wissen. Es wurden drei Arten probenbezogenen Wissens grob unterschieden.

In der Literatur existiert keine einheitliche Definition des Begriffs *Wissensmanagement*. Nach [Thobe 2003] ist es das zentrale Anliegen von Wissensmanagement, den Umgang mit der „wichtigen Ressource Wissen“ zu planen, sie organisiert zu vermehren und gezielt zu nutzen. Scheer (vgl. [Scheer 2002]) beschreibt die Aufgabe des Wissensmanagements als das Dokumentieren, Speichern, Nutzbarmachen und Erweitern der Wissensbasis. Koch (vgl. [Koch 1999]) sowie Probst (vgl. [Probst 1998]) nennen die folgenden Bausteine für Wissensmanagement: Wissensidentifikation, Wissenserwerb, Wissensentwicklung, Wissensverteilung, Wissensnutzung und Wissensbewahrung. Exemplarisch soll hier als ein Konzept des

Wissensmanagements das Prozeßmodell nach [Reinmann-Rothmeier 2000] beschrieben werden. In diesem Modell werden vier Prozeßkategorien des Wissensmanagements unterschieden:

- *Wissensrepräsentation* als die Menge aller Prozesse zur Identifizierung, Kodierung, Dokumentation und Speicherung bereits vorhandenen Wissens: Dies beinhaltet beispielsweise unter anderem das Strukturieren von Wissen (auch mit Hilfe von Probenerfassungsformularen) und seine Organisation in Form von (Proben-) Datenbanken (vgl. auch [Durst 2003]) oder Protokollen.
- *Wissenskommunikation* als die Menge aller Prozesse zur Vermittlung und Verteilung von Wissen: Dies umfaßt unter anderem Kommunikationskanäle wie Besprechungen und Trainings, aber auch die Nutzung verschiedener Kommunikationsmedien (beispielsweise Computernetzwerke).
- *Wissensgenerierung* als die Menge aller Prozesse, die neues Wissen hervorbringen: Im Biobanking umfaßt dies neben klassischen Prozessen wie externen Wissensbeschaffungen durch Neueinstellung von Mitarbeitern oder durch wissenschaftliche Kooperationen das Generieren von Wissen in wissensschaffenden Bereichen (beispielsweise durch Abarbeiten eines Protokolls zum Feststellen der Eigenschaften einer Probe). Prozesse der Wissenskommunikation zwischen Personen oder Institutionen tragen ebenfalls zur Schaffung von neuem Wissen bei, das durch Zusammenführung oder durch Interaktion einzelner Wissensbasen entsteht (vgl. Kombination in Abschnitt 2.1.3) Voraussetzung ist dabei die Kodierbarkeit bzw. Kommunizierbarkeit der Wissensinhalte.
- *Wissensnutzung* als die Menge aller Prozesse, die neu geschaffenes oder bekanntes Wissen anwenden: Wissen wird beispielsweise zu Handlungen, Dienstleistungen, Produkten oder Entscheidungen umgesetzt.

Die Ausführung von biomedizinischen Protokollen entspricht dem Prozeß der Wissensnutzung und ist gleichzeitig die Grundlage für den Prozeß der Wissensgenerierung. Dabei anfallende Dokumentation wird durch den Prozeß der Wissensrepräsentation akquiriert und kann auf verschiedene Arten gespeichert werden. Der Prozeß der Wissenskommunikation umfaßt Vorgänge wie das Abfragen von Probendaten oder den Austausch deklarativen Probenwissens.

Das folgende Kapitel 2.2 geht auf die Zusammenhänge zwischen Ausführung und Optimierung biomedizinischer Protokolle und den hier vorgestellten Aspekten von Wissenstheorie und Wissensmanagement ein.

2.2 Biomedizinische Protokolle und Kryokonservierung

Prozedurales Wissen zur Bearbeitung, Verwendung und Lagerung von Proben wird in biomedizinischen Forschungslaboratorien[47] in Form sogenannter *Protokolle* repräsentiert[48]. Dabei handelt es sich um listenartig strukturierte Arbeitsanweisungen an das Laborpersonal, deren Detailschärfe nicht definiert ist. Während manche Protokolle eine sehr detaillierte Vorgehensweise für jeden Prozeßschritt vorgeben, erfordern andere oftmals eine umfangreiche Substitution lückenhafter Prozedurbeschreibungen durch individuelles, eventuell fehlerhaftes, implizites Wissen (siehe Abschnitt 2.1.2). Das vorliegende Kapitel 2.2 beschreibt zunächst die Kryokonservierung als das zentrale Verfahren in Biobanking, aus dem sich unter anderem die benötigten Protokolle ergeben. Dann werden biomedizinische Protokolle aus Sicht des Wissensmanagements, aus Sicht der Graphentheorie und aus Sicht der Wirtschaftsinformatik betrachtet, um ihre Eigenschaften auf existierende Konzepte abbilden zu können und darauf basierend den eigenen Ansatz der vorliegenden Arbeit zu formulieren.

2.2.1 Kryokonservierung

Im Kontext von Biobanking wird im Besonderen auch prozedurales Wissen im Zusammenhang mit dem verwendeten Lagerverfahren benötigt. Das gegenwärtig einzig wissenschaftlich anerkannte Verfahren für die Langzeitlagerung lebender Zellen ist die sogenannte *Kryokonservierung*[49]. Abgeleitet vom altgriechischen Wort *kryos* (Eiseskälte, Eis) bezeichnet man damit ein Verfahren, das auf der langzeitkonservierenden Wirkung sehr tiefer Temperaturen (unter –130°C) auf lebende Zellen beruht. Solche Temperaturen halten vitale Zellen in einem biochemisch und physiologisch inaktiven Zustand (*Abiose*), so daß eine vitale Konservierung von Zellen über einen sehr langen Zeitraum möglich wird. Jedoch treten innerhalb des Temperaturbereiches zwischen 0°C und –40°C potentiell schädigende Effekte auf. Speziell das Wachstum von Eiskristallen innerhalb einer Zelle kann zu Schädigungen an der Plasmamembran, am Zytoskelett und an den Organellmembranen und dadurch letztlich zum Zelltod führen. Auch das Gefrieren extrazellulären Wassers und damit verbundene osmotische Effekte können irreversible Schäden an Zellen hervorrufen, denn damit verbunden sind eine Diffusion von Wasser aus der Zelle in den extrazellulären Bereich, eine Kontraktion

---

[47] Die Charakteristika biomedizinischer Forschungslaboratorien im Sinne der vorliegenden Arbeit werden in Kapitel 3.1 zusammengefaßt.

[48] Siehe auch den Aspekt der Wissensrepräsentation als Teil des Wissensmanagements in Abschnitt 2.1.4.

[49] Dieses Verfahren wurde erstmals bei der Konservierung von Spermatozoen angewandt (vgl. [Polge 1949]) und hat eine hohe Bedeutung in der Biomedizin erlangt. Etablierte Anwendungsbereiche sind *in-vitro-Fertilisation* (vgl. [Massip 2001], [Aisen 2002], [Leoni 2002]), Blutzellkonservierung (vgl. [Thomas 1996]) und Transport von Transplantaten (vgl. [Langer 1999], [Rajotte 1999], [Kuo 2002]).

der Zelle und eine Erhöhung der Konzentration gelöster Salze im Zytoplasma, woraus starke mechanische Belastungen entstehen. Unterhalb von –40°C findet jedoch anstelle eines Kristallwachstums eine Verglasung (*Vitrifikation*) von Wasser zu Eis statt. Die Schwierigkeit bei der Kryokonservierung besteht also hauptsächlich beim Einfrieren und Auftauen, da jeweils der kritische Temperaturbereich durchschritten werden muß.

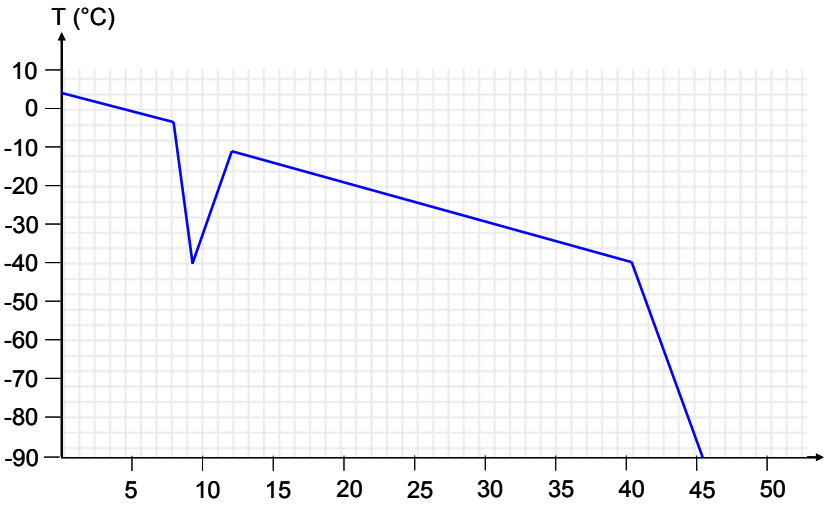

Abbildung 3: Typisches Temperaturprofil für einen Einfrierautomaten zum Einfrieren einer L929-Zellsuspension, optimiert auf 2ml Probenvolumen. Der Einfriervorgang beginnt bei 4°C. Die Proben werden schrittweise zunächst mit einer Rate von -1°C pro Minute bis auf -4°C abgekühlt, dann mit einer Rate von -25°C pro Minute bis -40°C abgekühlt, dann wieder bis auf -12 °C mit einer Rate von 10°C pro Minute erwärmt. Dann erfolgt erneutes Abkühlen mit -1°C pro Minute bis -40°C und anschließendes Kühlen bis -90°C pro Minute mit einer Rate von -10°C pro Minute. Im Anschluß erfolgt das Einlagern in einen Kryo-Lagertank.

Verschiedene Verfahren werden eingesetzt, um dem Eiskristallwachstums entgegen zu wirken. Genannt seien an dieser Stelle leichte osmotische Manipulationen der Zelle oder die Verwendung von *permeablen Kryoprotektoren* wie beispielsweise Glyzerin oder Dimethylsulfoxid (DMSO). Diese erniedrigen den Gefrierpunkt sowohl des intrazellulären wie auch des extrazellulären Wassers und vermindern durch geringere Eisbildung die Salzkonzentrationen innerhalb und außerhalb der Zelle[50]. Oftmals gelingen hinreichende Vitalitätsraten bei der Kryokonservierung von Zellen nur durch den Zusatz solcher Substanzen. Bei der Kryokonservierung tierischer Zellsuspensionen wird in der Regel 10% DMSO hinzugegeben. Um eine osmotische Belastung der Zellen weitgehend zu reduzieren, erfolgt die Zugabe sehr langsam und in kleinen Mengen. Überlebensraten zwischen 80% und 95% werden auf diese Weise erreicht. Ein bedeutender Nachteil von DMSO ist jedoch seine zellschädigende Wirkung bei Temperaturen über dem Gefrierpunkt von Wasser [Wusteman 2002], so daß insgesamt nur relativ geringe Konzentrationen dieser Substanzen benutzt werden können. Um Vitalitätsverluste durch die zytotoxische Wirkung von Kryoprotektoren

[50] Vgl. [Meryman 1971], [Muldrew 1999]

nach dem Auftauen zu reduzieren, ist ihr schnelles Auswaschen erforderlich[51]. Die Eignung verwendeter Kryoprotektoren und spezifisch optimierter Temperaturprofile (*Einfrier-* und *Auftaukurven*) kann anhand resultierender Überlebensraten bewertet werden. Überlebensraten werden mit Hilfe biochemischer Vitalitätstests bestimmt, die sowohl die Integrität der Plasmamembran als auch die Stoffwechselaktivität von Zellen nachweisen. Häufig werden zu diesem Zweck mehrere Fluoreszenzfarbstoffe in Kombination verwendet, die unter verschiedenen Voraussetzungen fluoreszieren. Ein Beispiel:

- *Fluoreszeindiazetat* ist ein Indikator für die Stoffwechselaktivität einer Zelle und durchdringt die Plasmamembran. Dort wird er durch den Zellstoffwechsel mit intrazellulären Esterasen zu Fluoreszein hydrolisiert, das im Fluoreszenzspektrum stark grün leuchtet und einen intakten Zellstoffwechsel und damit die Vitalität der Zelle anzeigt.
- *Ethidiumbromid* ist ein Indikator für geschädigte Zellmembranen, der durch eine Reaktion mit Desoxyribonucleinsäure (DNS) rot fluoresziert. Da Ethidiumbromid die Plasmamembran nicht durchdringen kann, ist ein Kontakt des Farbstoffs mit der DNS der Zelle nur bei beschädigter Membran und damit lethal geschädigter Zelle möglich.

Die Kombination beider Farbstoffe erlaubt das simultane Identifizieren vitaler Zellen anhand eines grünen Fluoreszenzeffektes und das Erkennen beschädigter Zellen durch einen roten Fluoreszenzeffekt. Folgende Abbildung zeigt einen solchen Vitalitätstest:

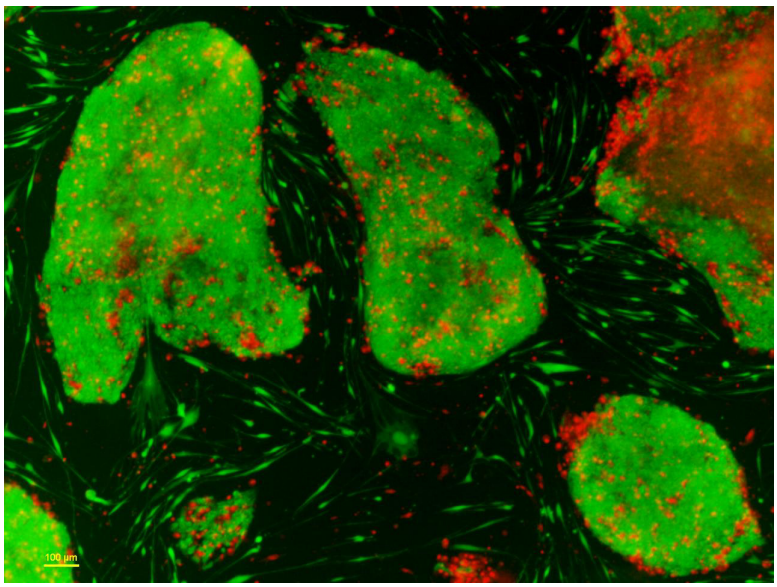

Abbildung 4: Fluoreszenzmikroskopische Aufnahme eines Vitalitätstests, angewandt auf Kolonien humaner embryonaler Stammzellen nach einer Kryokonservierung. Verwendet wurde eine Kombination aus den beiden Farbstoffen Fluoreszeindiazetat und Ethidiumbromid. Ein funktionaler Zellstoffwechsel führt zur Hydrolisierung von Fluoreszeindiazetat und somit zu einem grünen Leuchten. Dies ist ein Indikator für funktionale, vitale Zellen. Eine beschädigte Plasmamembran erlaubt das Eindringen von Ethidiumbromid in die Zelle und führt zu

[51] Vgl. [Sommerfeld 1999], [Wusteman 2002]

einem Kontakt mit der DNS, der durch rotes Leuchten erkennbar ist, das auf tote Zellen hinweist. Orangefarbene Zellen weisen beide Effekte auf und sind noch funktional, aber lethal beschädigt. Die Abbildung (zur Verfügung gestellt von Axel Beier) zeigt diese Effekte am Beispiel humaner embryonaler Stammzellen der Zell-Linie H1. Diese Stammzellen bilden sogenannte Kolonien (im Bild erkennbar als nahezu homogene Flächen), die von sogenannten *Feederzellen* versorgt werden. Die Feederzellen sind im Bild als einzelne, längliche Zellen erkennbar, die im Raum zwischen den Kolonien verteilt sind. Es handelt sich bei ihnen um murine embryonale Fibroblasten.

Das prozedurale Wissen im Biobanking umfaßt insgesamt Protokolle für:

- vorbereitende Präparationen, die vor dem eigentlichen Einfriervorgang ausgeführt werden müssen (beispielsweise Zugabe von DMSO)
- die Einfrierprozedur, anhand derer eine Probe gemäß Zelltyp-spezifisch optimiertem[52] Temperaturprofil abgekühlt werden muß, um die Probe in abiotischen Zustand zu versetzen
- eine Auftauprozedur, anhand derer eine Probe gemäß Zelltyp-optimiertem Temperaturprofil erwärmt werden muß
- weitere Präparationsschritte, die nach dem Auftauen der Probe anzuwenden sind (beispielsweise Auswaschen des Kryoprotektors, Bestimmen der Überlebensrate, analytische, diagnostische oder im Zusammenhang mit einer Zellkultivierung stehende Protokollschritte).

Das folgende Protokoll zum Einfrieren von hämatopoetischen Stammzellen mit Ficoll-Gradient ist ein Beispiel für die typische Repräsentation prozeduralen Wissens zur Vorbereitung von Zellen auf eine Kryolagerung:

- EDTA Blutröhrchen 8min bei 400 G ohne Bremse zentrifugieren
- Überstand fast abnehmen
- verbleibendes Blut auf 11ml auffüllen
- 4ml Ficoll in kleines Falcon Röhrchen vorlegen
- 11 ml Blut vorsichtig (Röhrchen schräg halten) auf Ficoll zupipettieren
- 25-30min bei 400g ohne Bremse oder kleinster Bremse zentrifugieren
- Serumüberstand mit 1000er Pipette abnehmen und verwerfen
- Ring (enthält die Stammzellen) mit der Pipette abnehmen
- in neues Röhrchen überführen
- Leukozyten-Ring in kaltem PBS resuspendieren (4°C) 5 ml PBS
- 8 min bei 400g bei 4°C zentrifugieren (Bremse aus)
- Überstand verwerfen
- Pellet in kaltem PBS resuspendieren

[52] Zelltyp-spezifische Einfrierprofile und Auftauprofile werden anhand empirischer Untersuchungen iterativ optimiert und sind das Resultat von kontinuierlichem Wissenszuwachs bzw. von organisationalem Lernen in einer wissenschaftlichen Gemeinschaft.

- 8 min bei 400g bei 4°C zentrifugieren (Bremse aus)
- Überstand verwerfen
- Zellen in RPMI-Medium mit mind 10% besser 20% autologem oder humanen AB-Serum (hitzeinakiviert) und 10% DMSO resuspendieren und einfrieren. Die Prozentangaben beziehen sich auf das Endvolumen.

Die konventionelle Kryokonservierung verwendet Kryoröhrchen für Probenvolumina zwischen 1ml und 10 ml Volumen. Ein Kryolagertank enthält mehrere Lagertürme, die in Schubladen unterteilt sind, die der Aufnahme der Kryoröhrchen dienen. In Kryolagertanks wird mit Hilfe flüssigen Stickstoffs eine tiefkalte, kryogene Gasphase erzeugt, deren Gradient von etwa –130°C im oberen Bereich bis hinab zu –196°C am Boden des Lagertanks reicht. Die Zuordnung von Daten zu Proben basiert auf referenzierenden Methoden unter Verwendung handschriftlicher Kennzeichnung oder Etiketten am Probengefäß.

Abbildung 5: Konventionelle Kryotechnologie. Oben links: konventionelle Probenröhrchen für die Kryokonservierung von Zellen, aufgesteckt auf einen Laborständer. Die Volumina der abgebildeten Röhrchen variieren zwischen 1ml und 5ml. Farbige Schraubverschlüsse können verwendet werden, um verschiedene Probentypen farblich zu kodieren. Oben rechts: eine Kryo-Schublade, teilweise bestückt mit Kryoröhrchen. Ein zusätzlicher transparenter Deckel mit aufgedruckten Positionsnummern erlaubt visuelle Lokalisierung und bietet Schutz gegen Herausfallen. Unten rechts: Die Schubladen werden in Kryo-Lagertürmen (*Racks*) gestapelt. Diese werden in Kryo-Lagertanks langzeitgelagert. Unten links: handelsübliche Kryo-Lagertanks in der Forschungs- und Demonstrationsbiobank der Fraunhofer-Gesellschaft in Sulzbach/Saar. Proben werden in Stickstoff-Gasphase

bei Temperaturen unter -130°C gelagert. Diese kryogenen Bedingungen werden durch Evaporation von flüssigem Stickstoff erreicht, der im unteren Bereich der Lagertanks vorgehalten wird. Die abgebildeten Lagertanks haben ein Volumen von jeweils 1400 Litern.

2.2.2 Biomedizinische Laborprotokolle aus Sicht des Wissensmanagements

Protokolle unterliegen einer iterativen Optimierung, deren Grundlage kontinuierlicher Wissenszuwachs ist. Dieser Wissenszuwachs entsteht einerseits durch die Nutzung des prozeduralen Wissens und durch Rückkopplung der Präparationsergebnisse auf das verwendete Protokoll, andererseits aber auch durch organisationales Lernen. Wissenszuwachs und iterative Optimierung von Protokollen auf Basis organisationalen Lernens können anhand der Spirale des Wissens und des SECI-Modells beschrieben werden:

- Das wiederholte praktische Ausführen eines existierenden, innerhalb einer Organisation oder wissenschaftlichen Gemeinschaft allgemein bekannten Laborprotokolls durch ein Individuum führt zu einer Verinnerlichung der einzelnen Handlungen (in Anlehnung an [Berry 1987], [Büssing 1999]). Diese Form unbewußten, impliziten Lernens durch Erfahrung in der praktischen Arbeitstätigkeit entspricht den Vorstellungen zum Aufbau und zur Weiterentwicklung impliziten Wissens durch Routinisierung bzw. Wissenskompilierung im Sinne von Andersons ACT-Modell [Anderson 1992]. In der Terminologie des SECI-Modells entspricht dies dem Vorgang der Internalisierung. Auf diese Weise entstehendes implizites Wissen fließt in künftiges erfahrungsgeleitetes Handeln ein.
- Der unbewußte Austausch individuellen, erfahrungsgeleiteten Wissens mit anderen Mitarbeitern im Labor, der durch Zusammenarbeit verschiedener Personen stattfinden kann, ergänzt oftmals die Internalisierung schriftlich zur Verfügung stehender Laborprotokolle. Ein Beispiel hierfür ist im Protokoll zur Isolierung mononukleärer Zellen aus Vollblut zu finden. Einer der Arbeitsschritte erfordert das Überschichten eines Lymphozyten-Separationsmediums mit einem Gemisch aus Blut und Pufferlösung. Das Protokoll beschreibt jedoch nicht die dabei notwendige Schräghaltung des Probenröhrchens[53]. Dieses implizite prozedurale Wissen wird zwischen Mitarbeitern implizit durch Beobachtung weitergegeben. In der Terminologie des SECI-Modells entspricht dies dem Vorgang der Sozialisation.
- Das schriftliche Dokumentieren ausgeführter Protokollschritte und verwendeter Reagenzien und das Dokumentieren von Ergebnissen (beispielsweise einer Vitalitätsrate) verbalisiert individuelles Wissen. Enthält die Dokumentation Details zu Protokollschritten, die im Protokoll selbst nicht enthalten waren, findet durch das Dokumentie-

[53] Dies ist ein Beispiel für die undefinierte Detailschärfe von Protokollen, die eine Substitution mit erfahrungsgeleitetem Wissen erfordert.

ren zusätzlich eine Externalisierung im Sinne des SECI-Modells statt. Das explizierte Wissen wird somit anderen Personen zugänglich, die dieses Wissen durch praktisches Anwenden internalisieren oder bewusst mit weiterem explizitem Wissen kombinieren, um Protokolle zu optimieren (*Learning Community*, vgl. [Mandl 2000]). Das Dokumentieren von Laborarbeit ist im Zusammenhang mit dem in Abschnitt 2.1.4 vorgestellten Prozeßmodell des Wissensmanagements die Voraussetzung für den Austausch von individuellem Wissen mit einer wissenschaftlichen Gemeinschaft.

- Die Optimierung bestehender Protokolle oder die Definition neuer Protokolle kann auch ausschließlich aus einer gemeinsamen expliziten Wissensbasis heraus erfolgen. Dies wird durch Korrelierung von Protokollen und ihrer Parameter mit explizit bekannten Effekten oder verbalisierten Resultaten erreicht. Diese Kombination im Sinne des SECI-Modells stellt der wissenschaftlichen Gemeinschaft korrigierte oder neue Protokolle zur Verfügung, die durch praktisches Ausführen wiederum internalisiert werden. Damit beginnt eine weitere Iteration.

Folgende Abbildung verdeutlicht diese Zusammenhänge mit Hilfe eines entsprechend konkretisierten SECI-Modells:

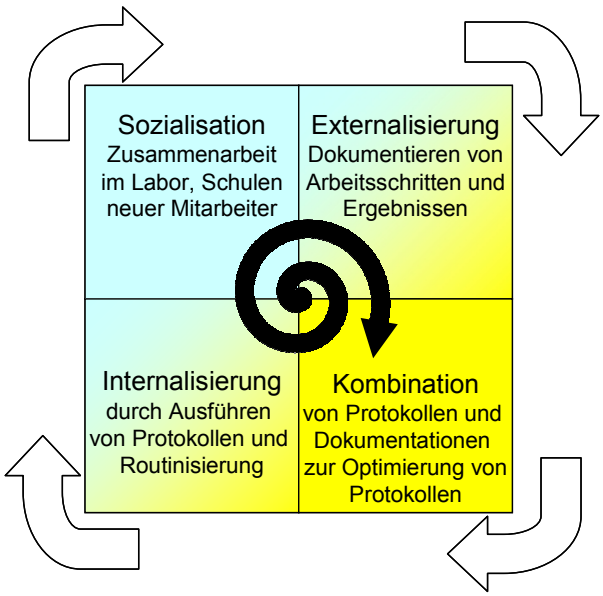

Abbildung 6: Konkretisiertes SECI-Modell zur Verdeutlichung von iterativer Protokolloptimierung und Austausch von individuellem Wissen mit einer wissenschaftlichen Gemeinschaft. Die Spirale des Wissens symbolisiert das iterative Durchlaufen der verschiedenen Arten der Wissensumwandlung. Sozialisation findet beispielsweise durch enge Zusammenarbeit im Labor oder durch Beobachtung statt[54]. Externalisierung impliziten proze-

[54] Ein Beispiel ist das implizite Lernen durch Beobachtung eines Überschichtungsvorgangs und seiner Details (Geschwindigkeit, Schräghaltung des Röhrchens). Die Personenkonstellation ist in biomedizinischen Laboratorien deutlich komplexer als in der klassischen Meister-Lehrling-Situation. Es ist auch unbewußte Wechselwirkung und gegenseitige Sozialisation zu vermuten, beispielsweise zwischen Technischen Assistenten und

duralen Wissens und Verbalisierung von Ergebnissen erfolgen durch schriftliche Labordokumentation, die die Wissensbasis einer wissenschaftlichen Gemeinschaft erweitert. Explizites prozedurales und deklaratives Wissen der Gemeinschaft ermöglichen die Optimierung und Definition von Protokollen durch Kombination. Die Ausführung von Protokollen führt zu einer Verinnerlichung expliziten prozeduralen Wissens und seiner Transformation in individuelles implizites Wissen.

Hinsichtlich Qualität der Protokollausführung und Qualität des dabei entstehenden deklarativen Wissens ist die erwähnte undefinierte Detailschärfe von Protokollen ein bedeutendes Problem. Die Substitution von fehlendem explizitem prozeduralem Detailwissen erfordert den Übergang in einen erfahrungsgeleiteten Handlungsmodus (vgl. [Büssing 2002]), der zwangsläufig Unterschiede bei der Ausführung per se identischer Protokolle durch verschiedene Personen verursacht. Konsequenz können Unterschiede im Präparationsergebnis sein, die im Nachhinein kaum mit den ursächlichen Abweichungen und Nuancen bei der Protokollausführung korreliert werden können. Zusätzlich von Bedeutung können Unterschiede bei der Qualität der Labordokumentation sein, da die Wissensakquise nicht im Detail vorgeschrieben ist und somit der Sorgfalt und individuellen Prioritäten unterliegt. Zusammenfassend können also weder eine akkurate Reproduzierbarkeit von Protokollausführungen noch eine zuverlässige Vergleichbarkeit entstehenden deklarativen Wissens angenommen werden.

2.2.3 Biomedizinische Laborprotokolle aus Sicht der Graphentheorie

Die einzelnen Präparationsschritte eines biomedizinischen Laborprotokolls können aufgefaßt werden als die Kanten $(i, j) \in E$ eines gerichteten Graphen $G = (V, E)$ mit $i, j \in V$. Ein Knoten $n \in V$ repräsentiert:

- die Ursprungsprobe, wenn der Eingangsgrad $d_G^-(n) = 0$ ist.
- Probenzwischenprodukte, wenn der Eingangsgrad $d_G^-(n) = 1$ und der Ausgangsgrad $d_G^+(n) > 0$ ist
- Probenprodukte, wenn der Eingangsgrad $d_G^-(n) = 1$ und der Ausgangsgrad $d_G^+(n) = 0$ ist.

Die Ausführung eines Protokolls entspricht einem Pfad von einer Ursprungsprobe zu einem Probenprodukt, also dem Pfad von der Wurzel zu einem Blatt des gerichteten Graphen.

Ein beliebiger Präparationsschritt T, repräsentiert durch Kante $(m, p) \in E$, transformiert eine Probe (repräsentiert durch $m \in V$) in eine gemäß Präparationsschritt T behandelte Probe (repräsentiert durch $p \in V$). Die folgende Abbildung zeigt dies für die beiden sequentiell aus-

---

Wissenschaftlern oder untereinander. Bei der Meister-Lehrling-Situation wird hingegen lediglich das implizite Wissen des Meisters unbewußt weitergegeben.

geführten Präparationsschritte T und W, repräsentiert durch die beiden Kanten (m, p) ∈ E und (p, q) ∈ E. Die Knotenmarkierung von m repräsentiert lediglich ein einziges Aliquot der Probe ‚907177':

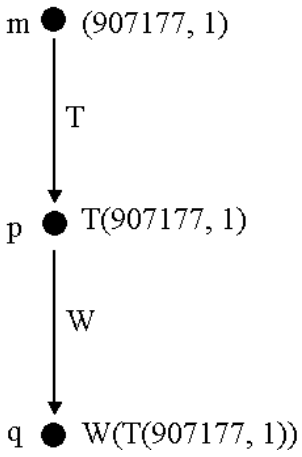

Abbildung 7: Sequentielle Bearbeitung einer Probe mit nur einem Aliquot durch zwei aufeinanderfolgende Präparationsschritte T und W, repräsentiert durch (m, p) ∈ E und (p, q) ∈ E mit m, p, q ∈ V.

Besteht ein Präparationsschritt A, repräsentiert durch eine Kante (g, h) ∈ E, in der Aliquotierung einer Probe in mehrere Fraktionen, dann repräsentiert der Knoten h die durch A entstandenen Aliquoten. Dies kann in Form eines Tupels (X, Y) erfolgen mit X:= ProbenID und Y:= Anzahl der Aliquoten von X. Im folgenden Beispiel wird das einzige Aliquot der Probe ‚907180' in acht Fraktionen aliquotiert:

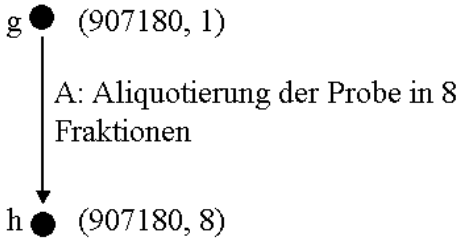

Abbildung 8: Aliquotierung der aus einem Aliquot bestehenden Probe ‚907180' durch Präparationsschritt A in acht Fraktionen, repräsentiert durch die gerichtete Kante (g, h) ∈ E mit g, h ∈ V.

Präparationsschritte biomedizinischer Laborprotokolle sind im Allgemeinen nicht auf eine einzige ProbenID beschränkt. Die gleichzeitige Präparation von Proben verschiedener Spender muß daher berücksichtigt werden. Sei F ein Präparationsschritt und sei (j, r) ∈ E mit j, r ∈ V eine gerichtete Kante, die F repräsentiert. Dann repräsentiert der Vorgängerknoten j die Menge S:= {(X, Y) mit X:= ProbenID und Y:= Anzahl der Aliquoten von X}. Der Nachfolgerknoten r repräsentiert die Menge S_F := {F(X, Y)} der gemäß Präparationsschritt F präparierten Elemente von S. Aliquoten S_F := {F(X, Y)} verschiedener Proben können auch verschiedenen parallelen Präparationsschritten B und C unterzogen werden, die zu unterschiedlichen Proben(zwischen)produkten S_{FB}:={B(F(X,Y))} und S_{FC}:={C(F(X,Y))} führen, repräsentiert durch Knoten u, v ∈ V, (r, u) ∈ E und (r, v) ∈ E. Folgendes Beispiel zeigt die Ab-

straktion einer Kombination aus Zentrifugation von Aliquoten mehrerer verschiedener Blutproben und anschließender Isolierung von Plasma (durch Präparationsschritt B) und Isolierung der Blutzellen (durch Präparationsschritt C)[55]:

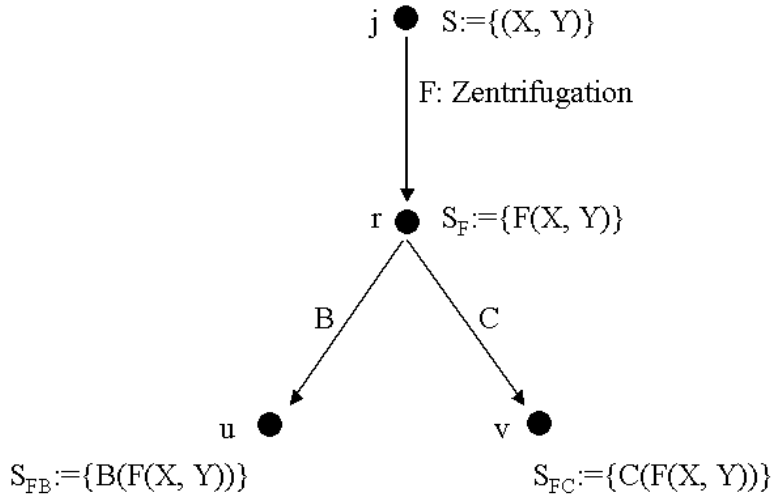

Abbildung 9: Zentrifugation der Aliquoten verschiedener Blutproben, repräsentiert durch S:= {(X,Y)}, in Präparationsschritt F, repräsentiert durch die gerichtete Kante (j, r) ∈ E mit j, r ∈ V, und anschließendes Isolieren von Plasma und Blutzellen durch die beiden parallelen Präparationsschritte B und C. (r, u) ∈ E mit r, u ∈ V repräsentiert Präparationsschritt B mit den resultierenden (Zwischen-) Produkten S_{FB}:={B(F(X,Y))}, und (r,v) ∈ E mit r, v ∈ V repräsentiert Präparationsschritt C mit den resultierenden (Zwischen-) Produkten S_{FC}:={C(F(X,Y))}.

Die Gesamtheit der Präparationsschritte e ∈ E überführt die Aliquoten der Ursprungsproben schrittweise in Zwischenprodukte und letztendlich in Aliquoten der gewünschten Produkte oder in wissenschaftliche Ergebnisse.

Die Ausführungen dieses Abschnitts sind im Zusammenhang mit dem Proben- und Aliquoten-Management des GHRC-Prototyps (beschrieben in Kapitel 5) von Bedeutung.

2.2.4 Biomedizinische Laborprotokolle aus wirtschaftsinformatischer Sicht

Die definierte Abfolge von Präparationsschritten, die durch biomedizinische Protokolle vorgegeben wird, entspricht aus wirtschaftsinformatischer Sicht dem Aufbau eines *Workflows*. Dieser Begriff bezeichnet eine vordefinierte Abfolge von *Aktivitäten* (*Activities*), die auf operative Aspekte fokussieren, beispielsweise auf die erforderlichen Parameter einer Zentrifugation. Jede Aktivität beschreibt eine Aufgabe, die als Teil eines Arbeitsablaufs zu

[55] Aufgrund unterschiedlicher Dichte von Blutplasma und Blutzellen befinden sich nach der Zentrifugation F das Blutplasma im oberen Abschnitt der Probenröhrchen und die Blutzellen im unteren Teil der Probenröhrchen. Schritt B dient dem Abnehmen des Plasmas, Schritt C dem Entnehmen der Blutzellen aus dem Zentrifugenröhrchen.

erledigen ist. Dabei handelt es sich um einen operativen Vorgang, dessen Ergebnis sowohl materieller Art (beispielsweise ein Probenerzeugnis) als auch informeller Art (beispielsweise eine festgestellte Vitalitätsrate) sein kann. Präparationsschritte können in diesem Sinn als Workflow-Aktivitäten aufgefasst werden. Im Gegensatz zum operativen Charakter eines Workflows beschreibt ein *Geschäftsprozess* die betriebswirtschaftlichen Aspekte (beispielsweise die Kosten) eines Arbeitsablaufs. Scheer [Scheer 2002a] bezeichnet Geschäftsprozesse als direkten Gegenstand betriebswirtschaftlicher Betrachtungen, für die Ziele wie etwa Verkürzung der Durchlaufzeit definiert werden können. Auch bezeichnet er sie als Gegenstand der Prozeßkostenrechnung. Als Intention zur Erstellung von *Geschäftsprozeßmodellen*[56] durch Abstraktion realer Prozesse wird von Scheer die Optimierung von Prozessen beispielsweise durch bessere Organisation von Unternehmenswissen genannt.

Geeignete Informationssysteme, die als *Workflow-Management-Systeme*[57] (*WMS* oder *WfMS*) bezeichnet werden, können die Ausführung von Workflows unterstützen. Sie führen eine *Workflow-Definition*[58], bestehend aus einzelnen elektronisch repräsentierten Activities, die in deterministischer Beziehung zueinander stehen, gemäß vordefinierter Algorithmen[59] aus. WMS werden beispielsweise nach [Scheer 2002a] eingesetzt, um zu bearbeitende Dokumente jeweils elektronisch zum entsprechend für den nächsten Bearbeitungsschritt benötigten Arbeitsplatz zu senden[60]. Der Bearbeitungsstatus der jeweiligen Aktivitäten ist dabei dem WMS bekannt. WMS sind allgemein definiert als Informationssysteme, die Instanzen einer Workflow-Definition nach einem vorgegebenen Schema abarbeiten und dabei erforderliche Daten sowie Applikationen zur Verfügung stellen. Ihre Aufgabe wird in der Koordination von Aktivitäten unter Berücksichtigung von Reihenfolgen gesehen und in der Verteilung von Rollen. Ihre Anwendung wird meist durch folgende Ziele motiviert:

- Vereinheitlichung von Arbeitsabläufen
- Verbesserung der Qualität von Arbeitsabläufen

---

[56] Eine XML-basierte Beschreibungssprache für Geschäftsprozeßmodelle ist beispielsweise die *Business Process Modeling Language* (*BPML*). Eine graphische Spezifikationssprache für die Modellierung von Geschäftsprozessen stellt die *Business Process Modelling Notation* (*BPMN*) dar.

[57] oder *Workflow-Systeme*

[58] Als *Workflow-Beschreibungssprachen* oder *Workflow-Definitionssprachen* werden Sprachen für die Erstellung operativ orientierter Definitionen von Arbeitsabläufen bezeichnet. Beispiele hierfür sind die XML-basierte *XML Process Definition Language* (*XPDL*, siehe auch Kapitel 5.5) und die auf Web-Services beruhende *Business Process Execution Language* (*BPEL*).

[59] Als Beispiele aus dem biomedizinischen Bereich seien *statische* und *dynamische Scheduling-Algorithmen* für die Verteilung von Ressourcen an Laborprozesse genannt.

[60] Bei diesem Anwendungsfall könnte beispielsweise der Dokumentenfluß einer Probenbestellung als Workflow-Definition vorgegeben und durch ein WMS abgearbeitet werden.

- Verkürzung von Bearbeitungszeiten
- Erhöhung von Informationsverfügbarkeit

Als Voraussetzung für die Steuerung von Workflow-Instanzen durch ein WMS nennt [Scheer 2002a] die Wohl-Strukturiertheit der Workflow-Definition, die durch vordefinierte Reihenfolge, Ablaufverzweigungen und zugeordnete Organisationseinheiten erreicht wird. Als Möglichkeiten, auch schwächer strukturierte Abläufe durch ein WMS auszuführen[61], wird die parallele Anordnung auszuführender Aktivitäten in der Workflow-Definition in Vebindung mit ad-hoc-Entscheidungen der Anwender angeführt.

Der Zusammenhang zwischen biomedizinischen Protokollen und Workflows (und entsprechend zwischen Präparationsschritten und Aktivitäten) ist die Grundlage für den in Kapitel 4.5 beschriebenen eigenen Ansatz und seine in Kapitel 5 beschriebene Umsetzung.

[61] Eine andere Möglichkeit der Unterstützung schwach strukturierter Abläufe durch ein Informationssystem ist die Verwendung von Groupware-Systemen, die jedoch keine logische Ablaufkenntnis besitzen.

3 Stand der Technik

3.1 Protokolle in Forschungslaboratorien: Ausführung und Dokumentation

Biomedizinische Forschungslaboratorien im Sinne der vorliegenden Dissertation zeichnen sich durch folgende Eigenschaften aus:

- Die Ausführung von Protokollen erfolgt durch Personen unter eventueller Einbeziehung von Laborgeräten und deren gerätespezifischen Software-Applikationen, falls vorhanden.
- Es können verschiedene Protokolle ausgeführt werden, auch simultan.
- Die Menge ausführbarer Protokolle eines Labors ist nicht definiert und jederzeit erweiterbar. Sie ist nur durch die Sicherheitsstufe des Labors, durch die verfügbare Ausstattung und personelle Kapazitäten beschränkt.
- Es existiert keine einheitliche Prozeßstrecke. Die verschiedenen Protokolle erfordern unterschiedliche Ressourcen, die in unterschiedlicher Zusammensetzung und Reihenfolge verwendet werden können.
- Sowohl die Ausführung bereits optimierter Protokolle wie auch die Entwicklung und iterative Optimierung von Protokollen können erfolgen.

Die Forschungslaboratorien des Fraunhofer-IBMT und die GHRC-Laboratorien in der Forschungs- und Demonstrationsbiobank der Fraunhofer-Gesellschaft in Sulzbach weisen diese Eigenschaften auf. Mehrere Protokollsammlungen stehen in der jeweiligen Forschungseinrichtung zur Verfügung. Zusätzlich existieren umfangreiche Online-Protokoll-Bibliotheken, die in der Terminologie des Wissensmanagements explizites prozedurales Wissens kommunizieren und einer größeren wissenschaftlichen Gemeinschaft zur Verfügung stellen. Wie in Abschnitt 2.2.2 beschrieben, kann das in Protokollen enthaltene explizite prozedurale Wissen nach dem SECI-Modell kombiniert, internalisiert, sozialisiert und externalisiert werden. Neben einer Optimierung durch iterative Wissensumwandlung ist durch Kombination expliziten Wissens auch das Ableiten neuer Protokolle möglich.

Die Auswahl von Protokollen für die Präparation von Proben oder für die Durchführung von Assays erfolgt durch Personen und spezifisch anhand des jeweiligen Probentyps. Ein ausgewähltes Protokoll wird in Papierform[62] ins Labor mitgenommen und dient dort als Vorlage während der Laborarbeit. Die folgende Abbildung zeigt ein Protokoll für die Isolierung mononukleärer Zellen aus Vollblut:

---

[62] Eine andere Methode ist das Verwenden von *QM-Software* wie beispielsweise ‚SOP-Speed'. Solche Produkte und ihre Anwendung wir weiter unten in diesem Kapitel erläutert.

Protokoll: Isolation mononukleärer Zellen aus Vollblut

Materialien:

- 1x PBS (ohne Mg^{2+}, Ca^{2+}) bei RT
- 1x PBS (ohne Mg^{2+}, Ca^{2+}) bei 4°C
- Eis

Vorbereitungen:

- Falkons bereitstellen
- Falkons mit Ficoll vorbereiten

Ausführung:

- Vollblut im Verhältnis 1:1 mit PBS (RT) in 50ml-Falkon verdünnen
- Mischen: 2-3x mit Stabpipette vorsichtig hoch und runter ziehen
- 20ml Ficoll / LSM Lymphocyte Separation Medium (PAA) mit 30ml Blut/PBS Gemisch vorsichtig und langsam überschichten
- für 30min, 2000rpm (~ 940 g), RT, ohne Bremse zentrifugieren (~ 45 min)
- nach der Dichtezentrifugation befinden sich im Röhrchen 4 Phasen:
  - gelbliche Phase: Plasma und Thrombozyten
  - schmale weißliche Bande: mononukleäre Zellen (Interphase)
  - farblose Bande: Ficoll
  - sedimentierte Erythrozyten und Granulozyten
- Zentrifuge sofort im Anschluss auf 4°C kühlen
- Interphase aller Röhrchen vorsichtig abnehmen und in 1-3 50ml-Falkons überführen
- Auffüllen mit PBS (4°C) auf 50ml
- 5min, 1200rpm, 4°C, mit Bremse zentrifugieren
- Überstand vorsichtig abnehmen und Pellet aufklopfen
- Auffüllen mit PBS (4°C) auf 50ml
- 5min, 1200rpm, 4°C mit Bremse zentrifugieren
- Überstand vorsichtig abnehmen und Pellet aufklopfen
- In 20 -50 ml PBS (4°C) resuspendieren
- Bestimmung der Zellzahl: CASY (Programm Monocyten)

Abbildung 10: Ein Protokoll zur Isolation mononukleärer Zellen aus Vollblut, zur Verfügung gestellt von PD Dr. Hagen von Briesen. Es listet zunächst die benötigten Reagenzien auf, die für die Protokollausführung benötigt werden, und gibt Auskunft über die vor Beginn der Isolationsprozedur zu treffenden Vorbereitungen. Dann folgt die Auflistung der Arbeitsschritte mit Angaben zu relevanten Geräteparametern und zu verwendenden Voreinstellungen für Geräteapplikationen.

Folgende Tabelle beschreibt auszugsweise die Ausführung des gezeigten Laborprotokolls durch eine Person unter Verwendung einer sterilen Laminar-Flow-Werkbank und verschiedener Laborgeräte. Die einzelnen Arbeitsschritte wurden während der Laborarbeit mit einer Videokamera aufgenommen[63] und anschließend mit ihren jeweiligen Arbeitsanweisun-

[63] Diese Videoaufnahme diente der Identifizierung impliziter Anforderungen an ein geeignetes Informationssystem. Die vollständige Protokollausführung und die dabei identifizierten impliziten Anforderungen an ein System werden in Abschnitt 5.3.3.4 beschrieben.

gen im Protokoll gegenüber gestellt. An dieser Stelle dient diese Gegenüberstellung zur Illustration der Bedeutung von implizitem Wissen bei der Ausführung von Laborprotokollen.

Anweisung	**Tatsächlicher Arbeitsschritt**
Vollblut im Verhältnis 1:1 mit PBS (RT) in 50ml-Falkon verdünnen	Acht Falcon-Zentrifugenröhrchen werden bereitgestellt. Jeweils etwa 20ml Lymphozyten-Separationsmedium ‚Ficoll' werden direkt aus der Flasche in die Röhrchen hineingeschüttet und die Röhrchen verschlossen. Der Blutbeutel mit Vollblut wird geöffnet. Das Blut (geschätztes Volumen 100ml) wird in eine Zellkulturflasche geschüttet. Das gleiche Volumen an PBS wird hinzugegeben.
Mischen: 2-3x mit Stabpipette vorsichtig hoch und runter ziehen	Die Zellkulturflasche wird verschlossen. Das Mischen erfolgt durch kreisende Handbewegung und ohne Pipette.
20ml Ficoll / LSM Lymphocyte Separation Medium (PAA) mit 30ml Blut/PBS Gemisch vorsichtig und langsam überschichten	Die vorbereiteten Falcon-Zentrifugenröhrchen werden geöffnet. Eine Automatikpipette wird auf langsamste Pipettiergeschwindigkeit eingestellt, um Vermischen der Phasen während des Überschichtens zu vermeiden. Die Zellkulturflasche mit dem Blut/PBS-Gemisch wird geöffnet und die Pipette mit jeweils 30ml des Gemischs befüllt, das langsam auf die innere Wand eines sehr schräg gehaltenen Falcon-Zentrifugenröhrchens abgegeben wird. Das Gemisch legt sich als dünner Film auf das Ficoll. Der Vorgang wird mit derselben Pipette für alle Röhrchen wiederholt. Die Zellkulturflasche wird während der ganzen Zeit bewegt, um ein Absetzen der Zellen zu verhindern.

Tabelle 3: auszugsweise Gegenüberstellung von Protokollpunkten und tatsächlicher Ausführung. Man sieht den Übergang in einen erfahrungsgeleiteten Arbeitsmodus zur Substitution fehlender Protokolldetails, denn die tatsächlichen Arbeitsschritte umfassen deutlich mehr Handlungen als in den jeweiligen Anweisungen formuliert sind.

Man sieht, daß individuelles Erfahrungswissen die lückenhaften Instruktionen des Protokolls substituiert hat. Dies entspricht dem Übergang in erfahrungsgeleitetes Arbeitshandeln, der in Abschnitt 2.1.2 nach [Büssing 2002] beschrieben wurde und Ursache für qualitative Unterschiede bei der Laborarbeit und ihrer Ergebnisse sein kann[64].

Die Dokumentationsakquise erfolgt in biomedizinischen Forschungslaboratorien handschriftlich[65] auf Papier. Sie soll das relevante Wissen aufzeichnen, das zur Charakterisierung der Protokollausführung und zum Beschreiben der Ergebnisse benötigt wird. Damit ist die resultierende Dokumentation von der individuellen Sorgfalt, persönlichen Prioritäten und subjektiven Beobachtungen geprägt und inhaltlich und somit qualitativ inhomogen. Eine einheit-

[64] Wie beschrieben, kann implizites Wissen auch fehlerhaft sein. Erfahrungsgeleitetes Arbeitshandeln auf Basis falschen impliziten Wissens kann in sicherheitskritischen Laboratorien zur Personengefährdung führen.

[65] Eine weitere Methode ist das Verwenden eines sogenannten elektronischen Laborbuchs. Methoden dieser Art werden weiter unten in diesem Kapitel beschrieben.

liche Qualität wäre jedoch erforderlich, um einerseits entstandenes Probenwissen miteinander vergleichen zu können, und um andererseits eine Protokollausführung genau reproduzieren zu können. Dies scheitert aber bereits an der fehlenden Akquise genauer Zeitpunkte und Zeitdauern bei der Protokollausführung, so dass auch biologische Effekte oder eintretende Veränderungen der Probe nicht mit zeitlichen Aspekten korreliert werden können. Ein Grund für die fehlende Akquise detaillierter Zeitpunkte ist darin zu sehen, dass die Akquise von Dokumentation oftmals erst nach Abschluß einer Protokollausführung begonnen wird, häufig wegen des Erfordernisses zügiger Bearbeitung, das sich aus der Empfindlichkeit von Proben ergibt. Der folgende Auszug aus einem Laborbuch verdeutlicht dies anhand der Dokumentation einer Ausführung des in Abbildung 10 gezeigten Protokolls für die Isolierung mononukleärer Zellen aus Vollblut:

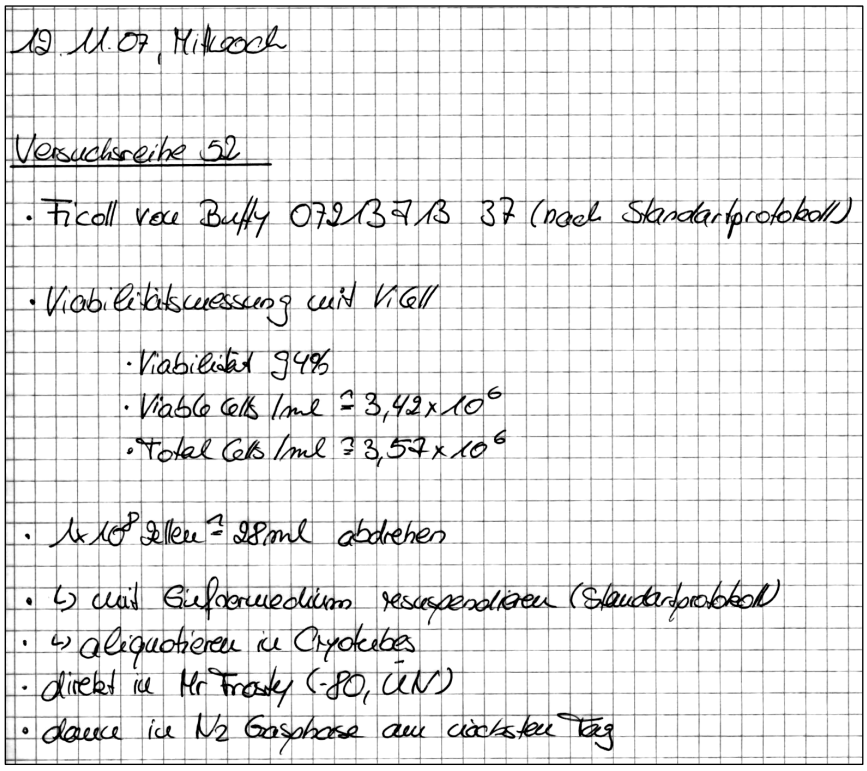

Abbildung 11: Auszug aus einem Laborbuch. Dieser Auszug dokumentiert eine Ausführung des oben beschriebenen Protokolls zur Isolierung mononukleärer Zellen aus Vollblut für Probe 0721371337. Akquiriert wurde ebenfalls entstandenes deklaratives Wissen in Form festgestellter Viabilitätsraten. Im Anschluß daran wurden mehrere Arbeitsschritte dokumentiert, die sich auf die Kryokonservierung der Probe beziehen.

Man erkennt, dass die Ausführung der im Protokoll aufgelisteten Schritte nicht einzeln dokumentiert ist. Vielmehr wird die Protokollausführung in einer einzigen Zeile der Dokumentation zusammengefasst. Auch Informationen zu verwendeten Reagenzien wurden nicht akquiriert. Wie oben geschildert, behindert diese fehlende Detailschärfe eine exakte Vergleichbarkeit von Ergebnissen. Reproduzierbarkeit der konkreten Protokollausführung ist auf Basis dieser Dokumentation nicht möglich. Im abgebildeten Laborbuchauszug wurde zusätzlich entstandenes deklaratives Wissen in Form von Viabilitätsraten akquiriert. Daran anschließend wurde die Ausführung eines weiteren Protokolls für die Resuspendierung mit Ein-

friermedium wiederum in einer einzigen Zeile dokumentiert. Schritte zur Kryokonservierung wurden hingegen in mehreren Zeilen dokumentiert. Insgesamt ist auch eine präzise Akquise von Zeitpunkten nicht erfolgt. Stattdessen wurden vage zeitliche Angaben notiert, die keine Korrelation biologischer Effekte mit zeitlichen Aspekten ermöglichen.

Im Rahmen von Bestrebungen nach Sicherung und Verbesserung der Qualität von Laborprozessen und ihrer Ergebnisse kommen in biomedizinischen Forschungslaboratorien auch Techniken und Systeme aus dem Bereich des *Qualitätsmanagements* (*QM*) zum Einsatz. Qualitätsmanagement bezeichnet aus wirtschaftswissenschaftlicher Sicht allgemein und über biomedizinische Bereiche hinaus alle organisierten Maßnahmen zur Verbesserung von Prozeß- und Produktqualität, zur Standardisierung von Arbeitsabläufen und zur Erhöhung der Prozeßeffizienz. Dies umfaßt auch die Standardisierung von Prozeßdokumentation. In Bezug auf Laborprotokolle sind aus der Sicht des Qualitätsmanagements sogenannte *Standard Operating Procedures* (*SOP*s) zu nennen. Es handelt sich um für eine bestimmte Aufgabe innerhalb einer Institution verbindliche Protokolle, deren Ausführung in SOP-spezifischen verbindlichen Formularen zu dokumentieren ist. Ziel ist, die Existenz verschiedener Protokoll- und Formularversionen für dieselbe Aufgabe zu vermeiden, um Qualitätsunterschiede von Laborprozessen, Dokumentation und Ergebnissen zu verringern.

Vor ihrer Anwendung durchlaufen SOPs und zugehörige Formulare Validierungs- und Genehmigungsverfahren und werden durch einen Auditor zertifiziert. Ihre Verwendung setzt somit einen gewissen durch Iteration erreichten Optimierungsgrad voraus. Gerade wegen des standardisierenden Charakters und der qualitätssichernden Intention dieser Dokumente sind auch QM-Maßnahmen für den Umgang mit solchen Dokumenten selbst erforderlich. Dazu zählen beispielsweise eine Kommunikationsstruktur zur Verteilung von SOPs und das Dokumentieren jeder Änderung an SOPs und Formularen (Techniken zum sogenannten *Audit-Trail*), aber auch das standardisierte Erstellen und Versionieren solcher Dokumente. Als QM-Systeme für solche Aufgaben werden im biomedizinischen Bereich häufig sogenannte *elektronische Dokumenten-Management-Systeme* (*EDMS*) eingesetzt. EDMS sind aus wirtschaftswissenschaftlicher Sicht Bestandteil von unternehmensweiten *Enterprise-Content-Management-Systemen* (*ECM*) und zeichnen sich im Wesentlichen durch folgende Merkmale aus:

- leichte Auffindbarkeit von Dokumenten durch verschiedene definierbare Indizes
- langfristige Lesbarkeit von Dokumenten durch Konvertiermöglichkeit in voraussichtlich langfristig kompatible Dateiformate
- Einhaltung gesetzlicher Archivierungsdauern
- Verwaltung verschiedener Versionen
- Integrierte Dokumentenerstellung
- Synchronisation bei Dokumentenerstellung durch mehrere Personen
- Integriertes Zugriffsberechtigungskonzept

- Protokollieren aller Änderungen und Manipulationen an den Dokumenten

Ein exemplarisches EDMS für den biomedizinischen Bereich ist die Applikation ‚SOP-Speed'. Es handelt sich um eine Software in Client-Server-Architektur, die erstellte SOPs und Formulare zentral archiviert. Ein integrierter Editor ermöglicht die Erstellung von Dokumenten und protokolliert relevante Bearbeitungsschritte. ‚SOP-Speed' erlaubt die Konvertierung der Dokumente in Standard-Formate, aber auch ihre elektronische Verteilung unter Nutzung eines proprietären Formats. Die folgende Abbildung veranschaulicht den Dokumentenfluß in EDMS, beginnend beim Entwurf eines Dokuments bis hin zu seiner Bereitstellung:

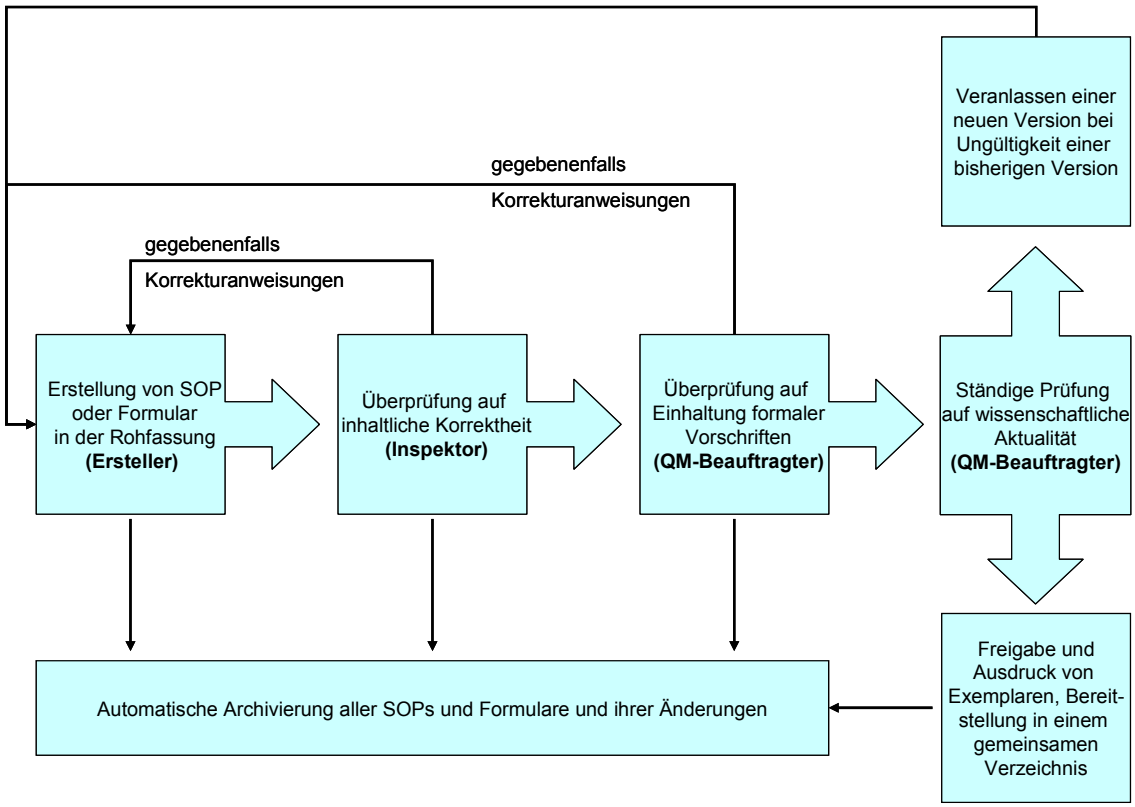

Abbildung 12: typischer Dokumentenfluß in EDMS, angelehnt an den Ablauf in ‚SOP-Speed'. Eine mehrstufige Validierungs- und Genehmigungshierarchie wird vor der Verwendung standardisierter Protokolle und Formulare durchlaufen. Die Rohfassung einer SOP oder eines Formulars wird durch einen Ersteller generiert und durch einen Inspektor auf inhaltliche Korrektheit überprüft. Sie wird anschließend entweder zur Durchführung von Korrekturen an den Ersteller zurückgeleitet oder an den QM-Beauftragten weitergeleitet, der auf Konformität mit formalen Vorschriften prüft und anschließend kontinuierlich die wissenschaftliche Aktualität des Dokuments überwacht. Es ist auch die Aufgabe des QM-Beauftragten, neue Versionen anzufordern und gültige Dokumente elektronisch und in Papierform zur Verfügung zu stellen. Sämtliche Änderungen an Dokumenten werden automatisch dokumentiert und archiviert (Audit Trail).

Eine Wissensakquise in die von ‚SOP-Speed' zur Verfügung gestellten Formulare ist nicht möglich. Somit ersetzt das System nicht das papiergebundene Dokumentieren von Protokollausführungen. Die Wissensakquise erfolgt unter Verwendung ausgedruckter Dokumentationsformulare. Im Zusammenhang mit potentieller Kontamination von Papier durch Krankheitserreger ist relevant zu erwähnen, dass papierbasierte Dokumentation in Laborato-

rien bestimmter Sicherheitsstufen[66] verbleiben muß und nicht ausgeführt werden darf. Da sie aber in der Regel zu Auswertungszwecken benötigt wird, hat sich in biomedizinischen Forschungslaboratorien die Behelfslösung etabliert, Papierdokumentation per Faxgerät an ein außerhalb der Laboratorien stehendes Faxgerät zu senden. Diese Fernkopie kann ausgewertet oder ihr Inhalt elektronisch erfasst werden. Grundsätzlich ist eine Transkription handschriftlicher Dokumentation in ein Informationssystem eine zusätzliche potentielle Fehlerquelle, denn unleserliche Handschrift, unklare oder unvollständige Angaben oder mangelnde Sorgfalt können zu Fehlern bei der Eingabe führen. Die meist geringere Qualität einer Fernkopie gegenüber dem Original-Dokument kann hier zusätzliche Schwierigkeiten verursachen.

Um der Problematik papiergebundener Dokumentation in Laboratorien zu begegnen, gibt es Ansätze, Labordokumentation nicht auf Papier, sondern direkt elektronisch zu erfassen. Dazu verwendete Applikationen werden als *elektronische Laborbücher* bezeichnet. Ihr Anwendungsbereich sind die automatische Speicherung und Analyse von Messwerten. Anhand dieser Funktionalität können sie den *Computer Aided Quality Assurance-Systemen* (*CAQ-Systeme*) zugeordnet werden, die häufig im Kontext von QM-Systemen Anwendung finden. Häufig sind sie Bestandteil von Systemen zur Laborautomatisierung, aber es existieren auch eigenständige elektronische Laborbücher, die eine rein manuelle Dateneingabe erfordern, beispielsweise das Web-basierte System ‚Elab'[67] des Hahn-Meitner-Instituts (HMI) Berlin [Fromme 2000]. ‚Elab' erlaubt das Erfassen von Daten mit Hilfe von HTML-Formularen und ihre zentrale Ablage unter Verwendung einer Oracle-Datenbank. Zur Verfügung stehende automatische Auswertungen umfassen beispielsweise die Erstellung von Graphen und ihre Darstellung in Koordinatensystemen. Die Eigenschaften der Benutzeroberflächen sind mit Hilfe eines Editors konfigurierbar. Ein Perl-Script stellt die Schnittstelle zur Datenbank dar. Die folgende Abbildung zeigt schematisch die Funktionsweise des Systems:

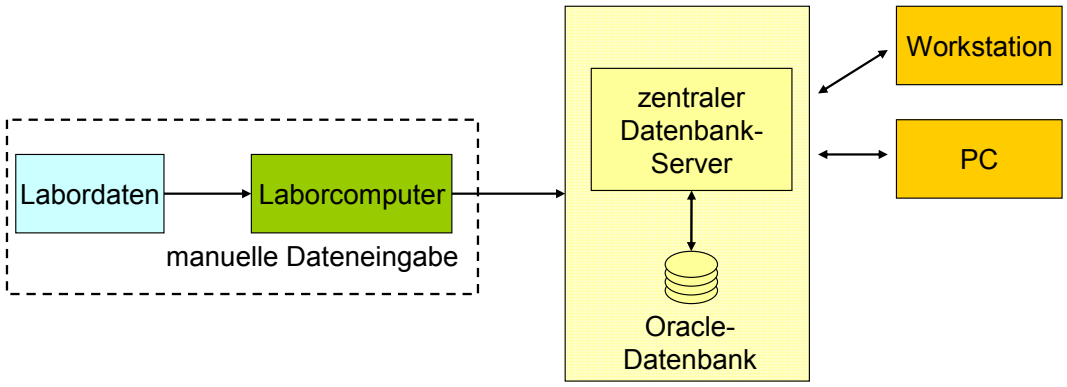

Abbildung 13: Schematische Darstellung des elektronischen Laborbuchs ‚Elab' des HMI Berlin. Labordaten werden mit Hilfe von HTML-Formularen manuell auf Laborcomputern erfasst und zentral in einer Oracle-

[66] Solche Laboratorien sind beispielsweise die GHRC-Laboratorien. Sie unterliegen der Sicherheitsstufe S3.

[67] Ein weiteres Beispiel für ein elektronisches Laborbuch, allerdings mit dem Fokus auf chemischen Versuchslaboratorien, ist das Produkt ‚ensochemLab'.

Datenbank gespeichert. Die Schnittstelle zur Datenbank wird durch ein Perl-Script realisiert. Workstations oder PCs außerhalb des Labors dienen der automatischen Auswertung oder Darstellung der erfassten Daten.

Die Fokussierung auf jeweils eine verbindliche Version eines Protokolls und des zugehörigen Dokumentationsformulars durch die Verwendung von SOPs und von EDMS reduziert Unterschiede bei Protokollausführung und Wissensakquise, die aus verschiedenen Versionen resultieren können. Elektronische Laborbücher können die Wissensakquise im Labor unterstützen und die Handhabung akquirierten Wissens vereinfachen. Die Kernprobleme in biomedizinischen Laboratorien bestehen aber weiterhin, nämlich (1) der Einfluß impliziten Wissens bei der Ausführung von Protokollen, (2) der Einfluß individueller Sorgfalt bei manueller Datenerfassung und (3) eine reale Gefahr von Fehlzuordnungen zwischen Probe, zugehörigem Protokoll und entstandener Labordokumentation durch die übliche referenzierende Zuordnungsmethodik.

3.2 Technologien zur Kennzeichnung und Identifizierung biologischer Proben

Das Spektrum an Behältern für biologische Proben ist sehr vielfältig und umfasst beispielsweise Titerplatten[68], Petrischalen oder Zellkulturflaschen, aber auch unterschiedlichste Probenröhrchen für die Verwendung der Probe im Labor oder zur Kryokonservierung. Entsprechend geeignete Kennzeichnungsmethoden müssen angewendet werden. Bei Blutbeuteln steht genügend Fläche zum Aufbringen eines Etiketts zur Verfügung, das neben einer reinen Probenidentifikation noch weitere Angaben enthält[69]. Bei Mikrotiterplatten oder Nanotiterplatten hingegen ist die Kennzeichnung jedes einzelnen Probenwells aus Platzgründen nicht möglich. Eine Identifizierung erfolgt deshalb über die Koordinaten der Probe auf der Titerplatte in Verbindung mit der Kennzeichnung der Titerplatte.

Bei Verwendung konventioneller Kryoröhrchen variiert das Volumen zwischen 1ml und 10ml. Diese Röhrchen werden in biomedizinischen Forschungslaboratorien[70] meist von Hand beschriftet. Das erfolgt entweder direkt auf dem Röhrchen oder unter Verwendung von Etiketten. Bei der Kryokonservierung ist zusätzlich eine grundsätzliche Unterscheidung von Probentypen durch Verwendung farbkodierter Schraubdeckel möglich.

---

[68] Titerplatten enthalten voneinander unabhängige Probentöpfchen in matrixförmiger Anordnung.

[69] Für die Kennzeichnung von Blutbeuteln gilt in Deutschland die ‚Empfehlung für die Behältnisbeschriftung von Blut und Blutzubereitungen'. Diese schreibt neben einer international eindeutigen Probennummer in Klartext und einer Darstellung in Barcodeform auch einen Produktcode vor, in dem Merkmale wie etwa Herkunftsland und Blutgruppe enthalten sind. Arzneimittelbezeichnung, Zulassungsnummer, Inhalt und Haltbarkeitsdatum sind weitere vorgeschriebene Angaben.

[70] gemäß der Definition aus Kapitel 3.1.

Bettendorf (vgl. [Bettendorf 2005]) beschreibt eine experimentelle Evaluierung von *Radio Frequency Identification Tags* (*RFID*) für die elektronische Kennzeichnung von Tumorproben. RFID-Lesegeräte ermöglichen die berührungslose Probenidentifikation durch Auslesen der im RFID-Tag gespeicherten Probenkennzeichnung. RFID-Technologie wird als mögliche Alternative zur handschriftlichen Probenkennzeichnung betrachtet. Die von Bettendorf beschriebene Integration von RFID-Tags im Schraubdeckel von Probenröhrchen stellt jedoch keine zuverlässige Probenidentifizierung dar, denn eine Verwechslung von Schraubdeckeln ist nicht auszuschließen[71].

Man kann die bisher existierenden Technologien zur Identifizierung von Proben wie folgt kategorisieren:

- Optische Kennzeichnung:
    - Handgeschriebene ID (nicht für Automatisierung geeignet)
    - Maschinenlesbare optische ID (beispielsweise Barcode)
- Nicht-optische Kennzeichnung (*smart tags*):
    - Mit Chip:
        - RFID (Chip +Antenne)
    - Ohne Chip:
        - EMID (elektromagnetische ID)
        - Magnetisches Wasserzeichen

Bei all diesen Verfahren handelt es sich um referenzierende Verfahren[72]. Probenbezogenes Wissen[73] ist nicht bei der Probe selbst gespeichert, sondern in Datenbanken oder Karteikartensystemen organisiert. Eine Ausnahme stellen Etiketten auf hinreichend großen Probenbehältern dar, wenn sie zusätzliche Minimalinformationen über die Probe enthalten. Im Regelfall wird das Wissen jedoch mit Hilfe der Probenkennzeichnung zugeordnet. Insbesondere im Kontext der Langzeitlagerung von Proben unter kryogenen Bedingungen und bei

---

[71] Das Fraunhofer-IBMT hat eine Möglichkeit entwickelt, RFID-Tags verwechselungssicher in Probenröhrchen zu integrieren. Dies wird in Kapitel 4.3 illustriert.

[72] Dieses referenzierende Verfahren und die beschriebenen Kennzeichnungen werden unter anderem von üblichen Probendatenbanken und Laborinformations-Management-Systemen genutzt.

[73] Siehe hierzu auch die drei im Biobanking identifizierten Wissensarten in Kapitel 2.1.

weltweitem Probenaustausch zwischen Laboratorien oder Biobanken sind der Verlust oder die Beschädigung einer Kennzeichnung realistische Risiken. In einem solchen Fall wird die Probe wertlos, denn das zugehörige Wissen kann nicht mehr zugeordnet werden.

3.3 Klassische Laborautomatisierung

3.3.1 Motivation

Die Aufgabe bioanalytischer Laboratorien wird in [Mole 1993] darin gesehen, qualitativ hochwertige Ergebnisse und Informationen innerhalb kurzer Zeit unter Einhaltung qualitätssichernder Richtlinien wie etwa GLP zu erzeugen. Um hohes Probenaufkommen zu bewältigen und Wettbewerbsvorteile zu erlangen, wird die Erhöhung der Produktivität bei Verringerung der Personalkosten angestrebt. Laborautomatisierung wird in diesem Zusammenhang als eine geeignete Lösung angesehen, denn Automatisierung in Unternehmen führt nach [Lindner 1990] zu effektiverer Forschung, verbesserter Produktqualität, höherer Produktivität und vermindertem Risiko für Personen. Bezogen auf analytische Laboratorien bedeutet eine Erhöhung von Produktivität eine höhere Anzahl getesteter Proben pro Zeiteinheit und damit eine Reduzierung der Kosten pro Probe. Automatisierte Systeme werden außerdem als Möglichkeit betrachtet, durch Personen verursachte Fehler bei gleichzeitig höherer Präzision und Akkuratheit auszuschließen, die Gefährdung von Personen durch chemische oder biologische Materialien zu reduzieren, kostenintensives Personal produktiver anderweitig zu beschäftigen und den Verbrauch von Proben und Reagenzien für Assays gegenüber manuellem Arbeiten zu minimieren.

Die Effekte von Laborautomatisierung auf die Produktivität eines klinischen Analyselabors über einen Zeitraum von 36 Jahren werden retrospektiv in [Sarkozi 2003] beschrieben. Der dort verwendete Produktivitätsindex (definiert als Anzahl der Testresultate pro Angestelltem pro Jahr) ist von 10600 (im Jahr 1965) auf 104558 (im Jahr 2000) angestiegen. Dies entspricht einem Faktor von 9,3. Gleichzeitig sind die Kosten pro Test von $0,79 auf $0,15 gesunken. Es wird geschildert, daß die Produktivität dieses Labors und die Anzahl der Proben kontinuierlich gesteigert wurde bei gleichzeitigem Abbau von Laborpersonal.

Die Definition von Laborautomatisierung nach [Mahaffey 1991] umfaßt einerseits den automatisierten Betrieb von Laborgeräten oder ganzen Laboratorien durch Laborautomatisierungssysteme und andererseits automatisiertes Informationsmanagement durch Laborinformations-Managementsysteme. Es gibt verschiedene Automatisierungsgrade, die im folgenden Abschnitt 3.3.2 beschrieben werden. Abschnitt 3.3.3 behandelt dann Laborinformations-Managementsysteme.

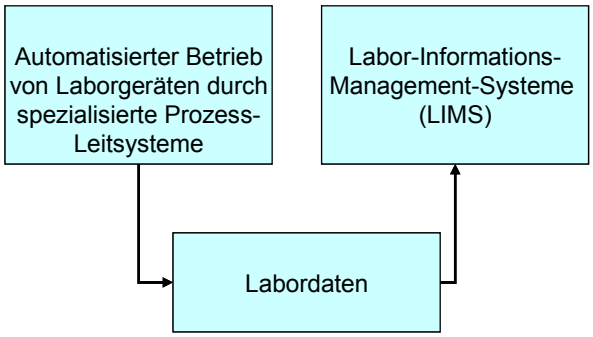

Abbildung 14: Laborautomatisierung nach [Mahaffey 1991]. Es wird einerseits zwischen dem automatisierten Betrieb von Laborgeräten oder ganzer Laboratorien und andererseits dem automatisierten Informationsmanagement unterschieden. Spezialisierte *Prozess-Leitsysteme* steuern den automatisierten Betrieb, und *Laborinformations-Managementsysteme* (*LIMS*) akquirieren die dabei anfallenden Daten, speichern sie in Datenbanken ab und werten sie aus. LIMS dienen oftmals auch der Verwaltung der Probendaten, zusätzlichen Fakturierungsaufgaben und dem *Customer Relationship Management*.

3.3.2 Laborautomatisierungssysteme

Laborautomatisierungssysteme sind spezialisierte *Prozessleitsysteme* (*PLS*). Ein Laborautomatisierungssystem dient zur Automatisierung von biotechnischen Laboratorien. Wie jedes Prozessleitsystem enthält ein solches System mindestens eine Anzeige- und Bedienkomponente. Diese ist in der Regel ein PC mit einer Laborautomatisierungssoftware (beispielsweise einer monolithischen Kontrollapplikation oder einem dynamischen Scheduler, siehe auch Kapitel 3.4) und dient zum Konfigurieren, Parametrieren, Bedienen und Überwachen des Systems. Laborgeräte sind über Schnittstellen an das System angeschlossen. Dazu dienen *prozessnahe Komponenten*, die den Anschluss von Laborgeräten, Robotern und Sensoren ermöglichen und die steuer- und regelungstechnischen Funktionalitäten ausführen. Als wichtigste Schnittstellen von Laborautomatisierungssystemen sind zu nennen:

- serielle Schnittstellen zum Anschluß von Laborgeräten (RS-232 oder RS-485, usw.)
- analoge Eingänge (beispielsweise für den Anschluß von Temperatursensoren)
- digitale Eingänge (für Grenzwertwächter)
- analoge Ausgänge
- digitale Ausgänge

3.3.2.1 Total Laboratory Automation (TLA)

Der Ansatz, Laboratorien vollständig zu automatisieren (‚human less laboratories'), hat zu teilweise sehr komplexen Laborautomatisierungssystemen geführt. Die Idee für die zunächst als *systematized automation* und später als *Total Laboratory Automation* (*TLA*)

bezeichnete vollständige Automatisierung entstand an der 1981[74] neu gegründeten ‚Kochi Medical School' in Japan [Sasaki 1998] aufgrund von Mangel an Laborpersonal und unbewältigbarem Probenaufkommen. Das von Sasaki entwickelte TLA-System wird auch als Fließbandsystem bezeichnet und basiert auf der Verwendung von Förderbändern und Robotern. Das Prinzip ist wie folgt: Probenröhrchen werden einzeln oder in Ständern auf das Fließband gesetzt und nacheinander in verschiedenen Analysegeräten und *Labormodulen* (siehe Abschnitt 3.3.2.2) bearbeitet und untersucht. Folgende Abbildung zeigt schematisch und allgemein die Funktionsweise von TLA aus der Vogelperspektive:

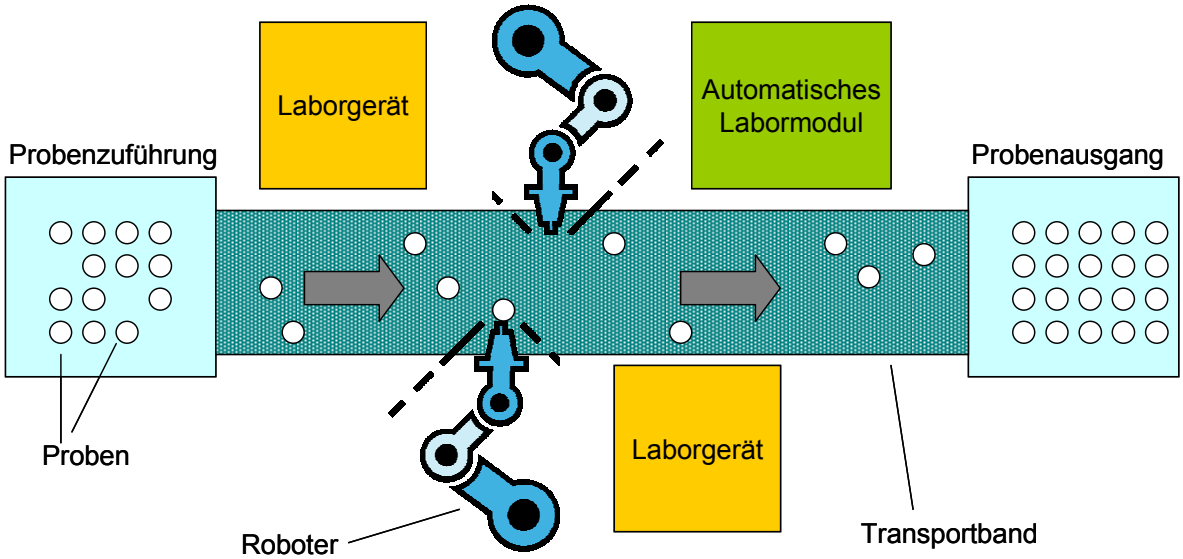

Abbildung 15: Schematische Abbildung eines TLA-Systems aus der Vogelperspektive. Es besteht aus einem Transportband, das die Proben von einer Probenzuführung bis hin zu einem Probenausgang transportiert. Pick-and-Place-Roboter nehmen die Proben zwischendurch vom Transportband und führen sie automatisierten Laborgeräten oder automatischen Labormodulen zu. Diese Geräte führen Bearbeitungs- oder Analyseschritte durch. Als Beispiele für Geräte seien Zentrifugen oder Inkubatoren genannt, als Labormodule seien Analysemodule genannt, die mit mehreren konsolidierten Laborgeräten ausgestattet sind (beispielsweise Pipettierroboter oder Fluoreszenzdetektor). Ein TLA-System ist stets eine kundenspezifisch maßgeschneiderte Lösung zur vollautomatisierten Ausführung der vor-analytischen und analytischen Aufgaben des spezifischen Laboratoriums.

Um 1990 wurden TLA kommerziell am Markt verfügbar. Daraufhin wurden bis 1998 TLA-Systeme in 72% der japanischen Universitätskliniken eingeführt. Ein großes Problem bei der Umsetzung von TLA sind die sehr hohen Investitionskosten, denn ein TLA ist eine Kunden- und Labor-spezifisch entworfene Kombination aus Geräten und Transportmechanismen [Felder 1998]. Eine bedeutende Schwierigkeit bei der Umsetzung von TLA ist die nicht genügende Standardisierung von Laborgeräten und Förderbändern, die oftmals Modifikationen an Geräten erfordert, um eine Integration zu erreichen. In [Sasaki 1998] wird unter ande-

[74] Zeitgleich begann die Entwicklung automatischer Laborgeräte durch Hersteller wie Hitachi, Toshiba oder Olympus.

rem eine Standardisierung von Laborgeräten, Lagertechnologie und darin integrierter Robotik gefordert, um TLA nach einem ‚Plug-and-Play'-Prinzip aufbauen zu können. Hoffmann [Hoffmann 1998] schätzt die Gesamtanzahl von TLA-Systemen weltweit auf etwa 900.

Folgende Abbildung zeigt schematisch und vereinfacht das von Sasaki entwickelte und in [Sasaki 1998] beschriebene TLA-System:

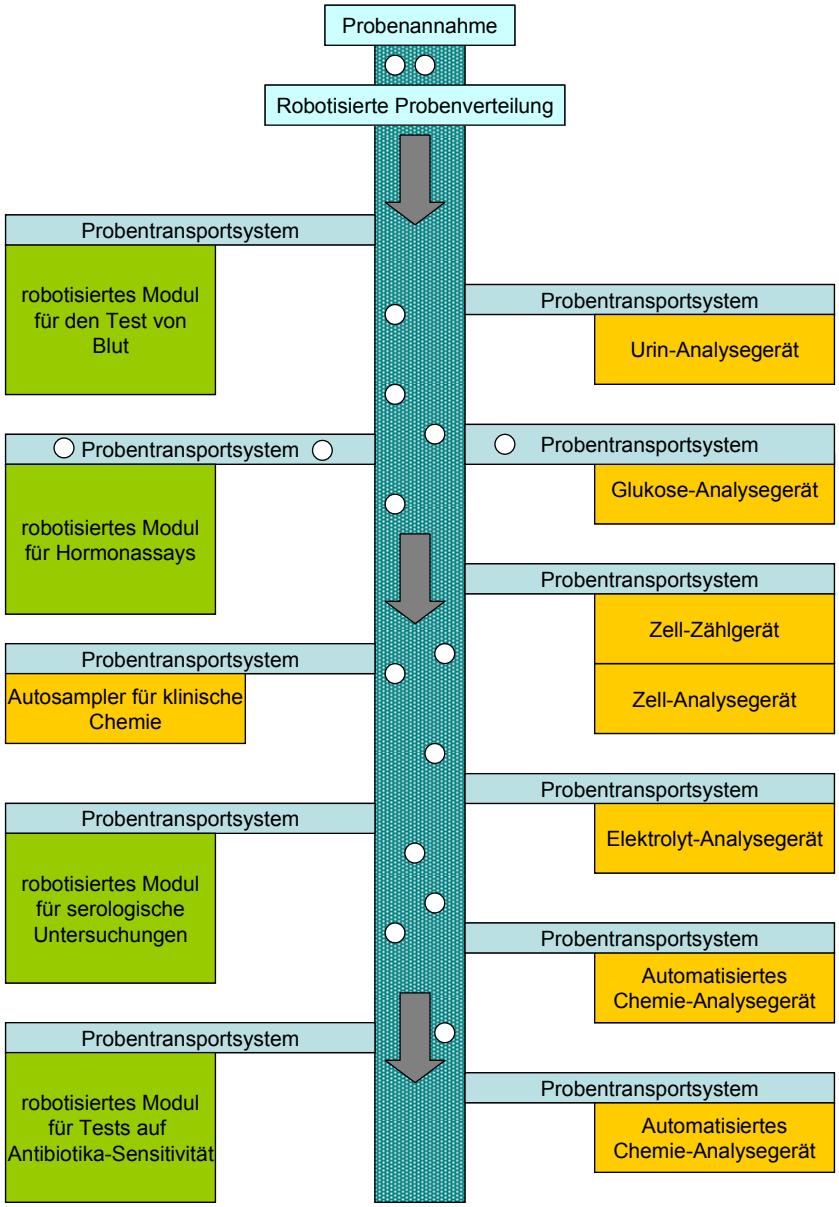

Abbildung 16: Schematische und vereinfachte Darstellung des in [Sasaki 1998] beschriebenen Förderband-Systems der ‚Kochi Medical School'. Eine robotisierte Probenverteilung in Verbindung mit einem Hauptförderband und untergeordneten Probentransportsystemen erlaubt das gezielte Zuführen von Proben zu benötigten Laborgeräten (orange dargestellt) und automatisierten bzw. robotisierten Labormodulen (grün dargestellt). Das Hauptförderband verläuft in einer Höhe von 2,40m. Daher sind Aufzugssysteme erforderlich, die in dieser Abbildung als Teil des jeweiligen Probentransportsystems zwischen Förderband und Gerät dargestellt sind.

Ein ebenfalls vollautomatisiertes Laborsystem, jedoch mit dem Fokus auf der Analyse von Wasserproben, wird in [Wyatt 1996] beschrieben. Dieses System ist unterteilt in einen

TLA-Teil für die Vorbereitung der Analyse und einen separaten TLA-Teil für die eigentliche Analyse. Der Transport zwischen beiden Systemteilen wird von Personen durchgeführt. Dieses System besteht unter anderem aus 22 Robotern, 21 lokalen Computern, 5 Steuerrechnern und einem LIMS.

Eine Alternative zum Förderband in TLA-Systemen wird in [Gentsch 1993] gezeigt. Dort wird der Transport von Proben zwischen einzelnen Laborgeräten oder gar einzelnen Laboren mit Hilfe eines auf Schienen befindlichen Transportroboters realisiert (siehe folgende Abbildung):

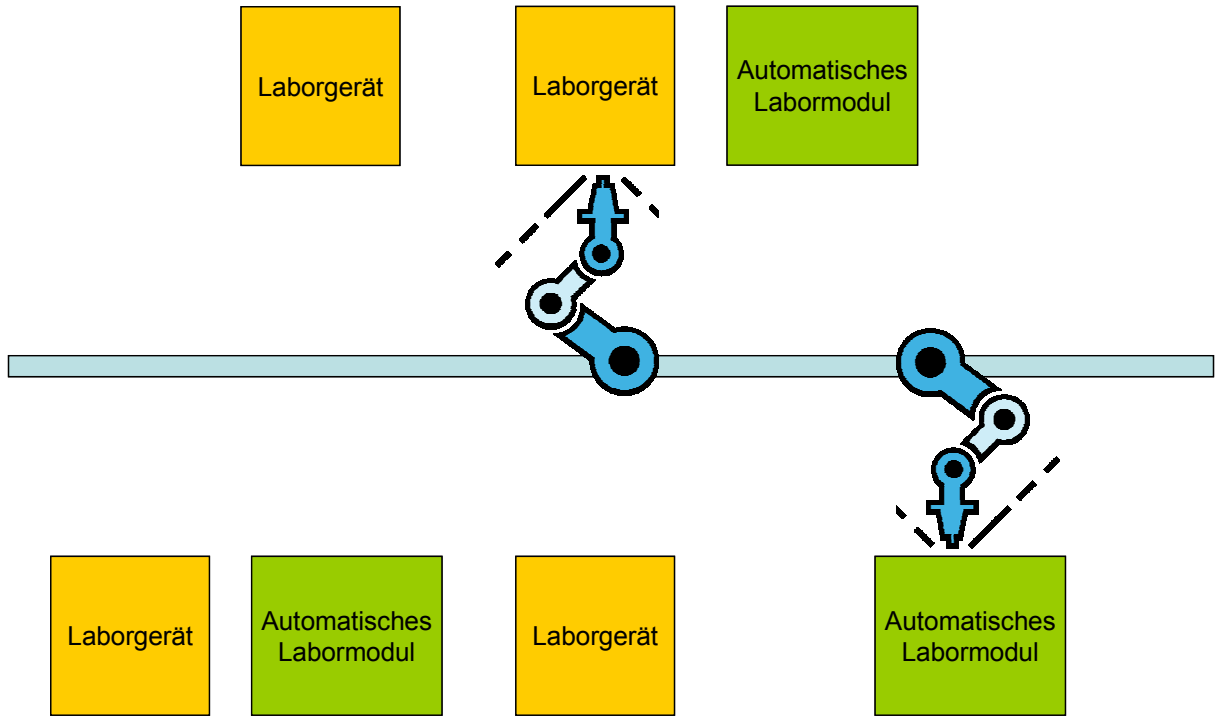

Abbildung 17: Schematische Abbildung eines flexiblen Transports durch schienengeführten Roboter, angelehnt an [Gentsch 1993]. Die Roboter können nach Bedarf die benötigten Positionen zur Bestückung der einzelnen Laborgeräte und Module anfahren.

In [Tatsumi 1999] werden auf Basis mehrjähriger Verwendung von TLA-Systemen in Laboratorien Vor- und Nachteile geschildert. Als Vorteile von TLA-Systemen werden genannt:

- signifikante Kosteneinsparung
- High-Speed-Bearbeitung auch großer Probenanzahlen
- definierte Bearbeitungszeit
- Verbesserung der Präzision durch Reduzierung menschlicher Fehler
- einfache Handhabung von Test- und Report-Systemen
- sauberes und sicheres Arbeiten

Als Nachteile und Probleme von TLA-Systemen werden genannt:

- Verlust an beruflicher Qualifikation und Fähigkeit beim Laborpersonal
- Sehr hohe Ausgaben für Installation und Bereitstellung von Schnittstellen bei Integration neuer Laborgeräte
- Unflexibilität von TLA-Systemen

Yavilevich (vgl. [Yavilevich 2002]) faßt Probleme zusammen, die bei dem Versuch auftraten, TLA für bioanalytische Laboratorien in den USA einzuführen. Er führt diesen Mißerfolg auf die wesentlich geringere Bevölkerungsdichte in USA gegenüber Japan zurück, die in einer deutlich geringeren Anzahl zu testender Proben resultiert.

3.3.2.2 Modulare Laborautomatisierung

Felder (vgl. [Felder 1998]) nennt Laborautomation als unverzichtbaren wirtschaftlichen Faktor im Wettbewerb zwischen Laboratorien. Gleichzeitig beschreibt er, daß schätzungsweise nur etwa 8% aller US-amerikanischen Laboratorien die hohen Kosten für die Umsetzung von TLA-Systemen aufwenden könnten. Dieser Problematik widmet sich die sogenannte *modulare Laborautomatisierung*. Darunter versteht man den Einsatz dezentraler automatisierter Module, die auf die Ausführung einer bestimmten Klasse von Arbeitsabläufen spezialisiert sind. Module bestehen aus automatisierten Einzelgeräten, die miteinander kooperieren, beispielsweise Barcodescanner, Inkubatoren, Pipettierarrays und Fluoreszenzreader. Der Transport von Proben zwischen diesen einzelnen Geräten kann beispielsweise durch einen kartesischen Roboter erfolgen, der ebenfalls in dem Modul verbaut ist. In [Felder 1998] werden im Kontext modularer Automatisierung unter anderem folgende Begriffe definiert:

- *analytische Konsolidierung*: der Vorgang, mehrere Analysetechniken in einem Gerät zu kombinieren
- *konsolidiertes Gerät*: ein Analysegerät, das eine Vielzahl analytischer Techniken beherrscht
- *Specimen manager*: ein mechanisches Gerät, in dem Proben vor und nach ihrer Analyse aufbewahrt werden können
- *Workcell*: eine Kombination aus Specimen manager und (konsolidierten) Geräten, die nicht als integrierte Stand-alone-Lösung ausgeführt ist
- *Modular workcell*: eine Kombination aus Specimen manager und (konsolidierten) Geräten, die voll integriert ist und als Stand-alone-Lösung arbeitet

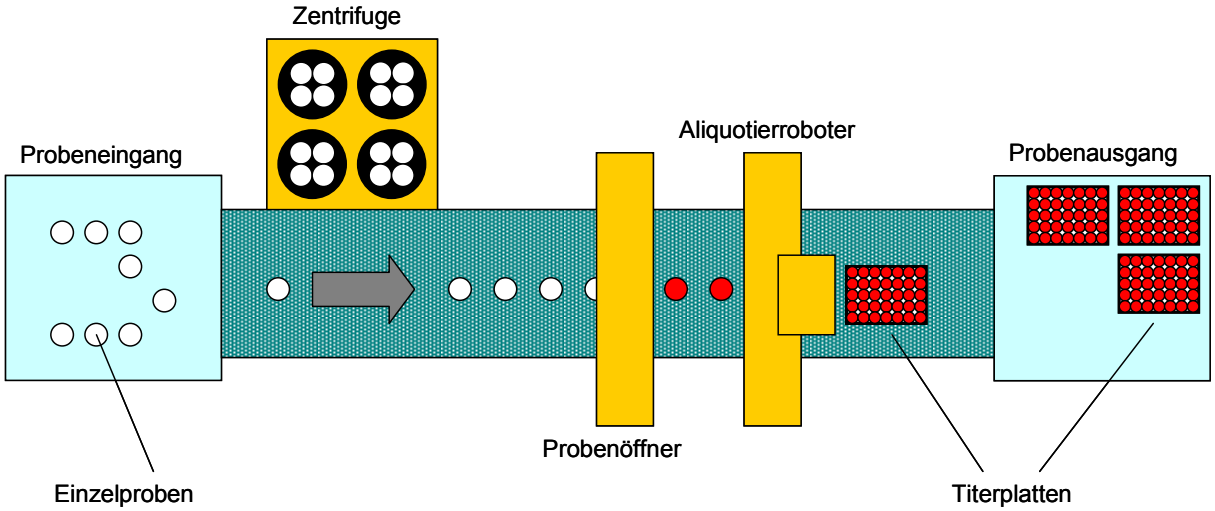

Abbildung 18: Schema eines integrierten automatischen Labormoduls (Modular Workcell) für vor-analytische Präparationsprotokolle. Proben werden zentrifugiert, geöffnet und aliquotiert. Jeweils ein Probenpuffer hält vorzubereitende Proben in Probenröhrchen bzw. zur Analyse vorbereite und aliquotierte Proben auf Titerplatten vor.

Neben den wesentlich geringeren Anschaffungskosten gegenüber TLA ist als weiterer Vorteil modularer Automatisierung der deutlich geringere Platzbedarf automatischer Labormodule zu nennen[75]. Dies wird anhand folgender Abbildung deutlich:

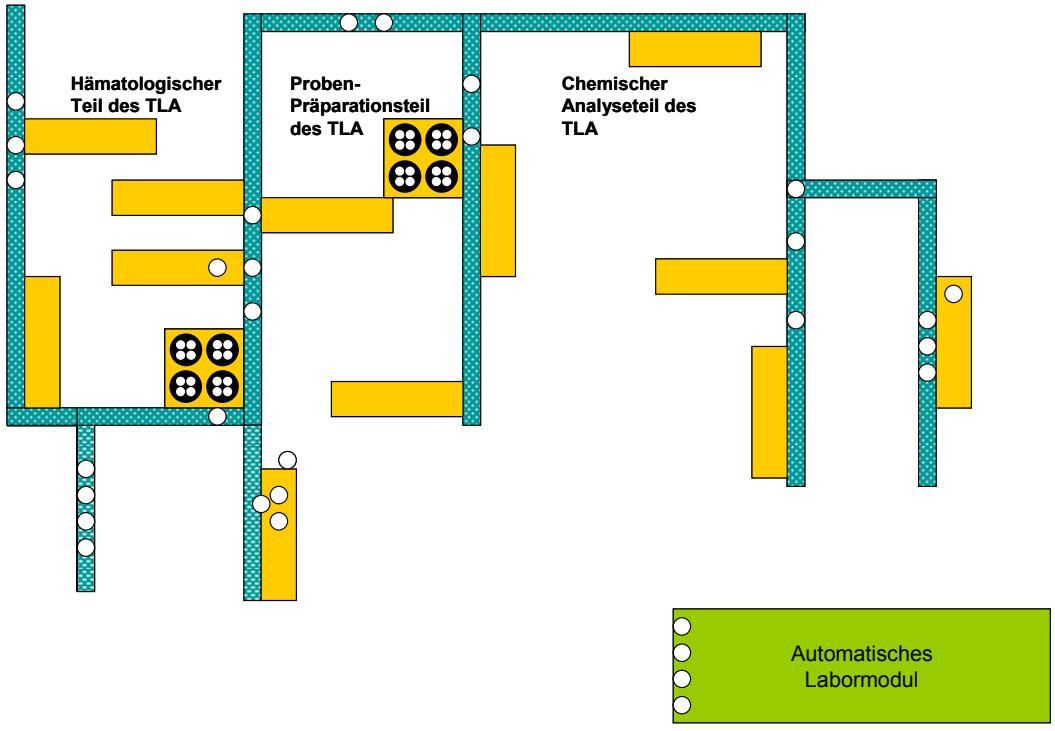

Abbildung 19: Schematischer Größenvergleich zwischen einem TLA und einem automatischen Labormodul (Modular Workcell), angelehnt an eine Abbildung aus [Felder 1998]. Die orangefarben dargestellten Laborgeräte sind durch ein oder mehrere Förderbänder (grün dargestellt) verbunden. Die Proben (weiß dargestellt) werden

[75] Hinsichtlich der Flexibilität bei Änderungen oder Erweiterungen an der Geräteausstattung besteht jedoch kein grundsätzlicher Vorteil von modularer Automatisierung gegenüber TLA-Systemen. Dieser Aspekt ist von der Architektur der verwendeten Laborautomatisierungssoftware abhängig. Siehe dazu Kapitel 3.4.

zwischen den verschiedenen Laborgeräten, beispielsweise Zentrifugen, transportiert. Die beiden abgebildeten Automatisierungslösungen haben gleiche Performanz. Der deutlich höhere Platzbedarf des TLA motiviert die Verwendung hochspezialisierter, platzsparender Module.

Modulare Automation wird auch im High-Throughput-Screening in der Pharmazie auf der Suche nach relevanten Wirkstoffen eingesetzt. Folgende Abbildung zeigt das *Assay Screening Modul* ‚SCARINA' in Verbindung mit dem *Micro to Nano Reformatting Modul* ‚MITONA' (beide: Evotec Technologies GmbH, Hamburg) zur Illustration automatischer Labormodule:

Abbildung 20: Illustration automatischer Labormodule. Abgebildet sind das *Assay Screening Modul* ‚SCARINA' (rechts, Vordergrund) und das *Micro to Nano Reformatting Modul* ‚MITONA' (links, Hintergrund), entwickelt von Evotec Technologies GmbH, Hamburg. Beide Module sind zusammenfassend in [Gentsch 2000] beschrieben. ‚SCARINA' dient dem pharmazeutischen Hochdurchsatz-Wirkstoffscreening. Zu testende Substanzen werden mit benötigten Reagenzien versehen und mit Hilfe eines auf konfokaler Fluoreszenzmessung beruhenden Detektionssystems vermessen und ausgewertet. Dieses Verfahren arbeitet mit Volumina im Nanoliterbereich und beruht auf Einzelmolekülnachweis. ‚MITONA' dient dem vollautomatischen Umfüllen von Proben aus Mikrotiterplatten in Nanotiterplatten als Voraussetzung für das Screening durch ‚SCARINA'. Die abgebildeten Module werden mit Hilfe geeigneter Geräteapplikationen betrieben und von einem *System Controller*[76] gesteuert. Aspekte von Labor-Automatisierungssoftware werden in Kapitel 3.4 betrachtet.

[76] Definition siehe Kapitel 3.4. Die Geräte der beiden abgebildeten Module werden mit Hilfe des Schedulers ‚Bernstein' gesteuert, der ausführlich in Kapitel 5 diskutiert wird.

3.3.3 Laborinformations-Managementsysteme

Dieser Abschnitt widmet sich dem Aspekt des automatisierten Informationsmanagements, das nach [Mahaffey 1991] weiterer Bestandteil der Laborautomatisierung ist und sich komplementär zu Laborautomatisierungssystemen verhält. Die Akquise deklarativen Wissens über Laborautomatisierungssysteme und deklarativen Wissens, das durch diese Systeme erzeugt wird, ist nach [Whelan 2004] die Aufgabe von *Laborinformations-Management-Systemen* (*LIMS*). Muller (vgl. [Muller 1999]) führt zusätzlich das Management von Proben als weitere Aufgabe solcher Systeme an. Gibbon (vgl. [Gibbon 1984]) betrachtet LIMS als ein „outgrowth" von Laborautomatisierung und definiert sie als „computerisierte Systeme zum Informieren über ein analytisches Labor und alle in ihm vorhandenen Proben, ihren aktuellen Standort und Präparationsstatus"[77]. Wood (vgl. [Wood 1993]) beschreibt die Aufgabe von LIMS als „to integrate instrumentation and data handling and to provide easy and quick distribution of the reported data to a large network of clients". Er sieht LIMS als idealen Ausgangspunkt für das papierlose Labor an[78] und verweist in diesem Zusammenhang auf die Richtlinien zur *Good Automated Laboratory Practice* (*GALP*).

Unterschiedlichste Laborgeräte können in LIMS integriert werden. Dazu erforderlich ist eine geeignete Schnittstelle, über die das jeweilige Gerät mit einem Laborcomputer verbunden werden kann, um die anfallenden Geräte- oder Analysedaten zu übertragen. Diese Daten werden in benötigte Reportformate eingearbeitet [Gillespie 1997], ausgewertet oder in elektronische Laborbücher[79] übertragen. Die Erfassung der Daten wird auch als Meßstrang, die Verarbeitung und Auswertung der Daten auch als Kontrollstrang bezeichnet. LIMS leisten nach [Cagindi 2004] einen wesentlichen Beitrag zur Qualitätssicherung und damit zur Effizienz eines Laboratoriums und tragen zur Reproduzierbarkeit von Protokollausführungen bei, indem sie auch Metadaten[80] (beispielsweise gerätespezifische Parameter) aufzeichnen. Daten und Metadaten werden bei LIMS in einer zentralen Datenbank gespeichert. Dazu ist ein Datenbankmodell erforderlich, das unter anderem die Struktur der zu dokumentierenden Prozeßkette abbildet und die zu erfassenden Daten definiert. Dementsprechend können LIMS lückenlos Prozeßketten dokumentieren, beginnend bei der Probenregistrierung über die Kontrolle und Dokumentation der Laborprozesse bis hin zur Auswertung der akquirierten Daten.

---

[77] Englischer Originalwortlaut: "computerized system designed to provide on-line information about a support/ service analytical laboratory and the samples being analysed in the laboratory. The information provided includes the current location of all samples in the laboratory, along with the status of all analyses (i.e. not begun, in progress, awaiting approval, and completed)."

[78] Siehe auch die Ausführungen in Kapitel 3.1 zur Problematik papiergebundener Dokumentation.

[79] Siehe auch die Ausführungen in Kapitel 3.1 zu elektronischen Laborbüchern.

[80] Gerätespezifische Parameter entsprechen prozeduralem Wissen.

Als grundsätzliche Funktionalitäten von LIMS nennt [Cagindi 2004]:

- *Gerätemanagement*: zentralisierte Speicherung von Wartungs- und Kalibrierungs-Logs in der LIMS-Datenbank
- *Datenmanagement*: Abbildung der gesamten Laborstruktur in der Datenbank, um Informationen über Laborpersonal, Geräte, Analysemethoden, Arbeitsabläufe und Kosten zu organisieren
- *Validierung*: Prüfen von Datenintegrität anhand integrierter Validierungstechniken
- *Probenmanagement*: Techniken für die Registrierung, das Beobachten des Präparationsfortschritts, Autorisierung und Archivierung von Proben und Reagenzien
- *Ressourcenmanagement*: Zeiterfassungsfunktionen, Kostenberechnung
- *Kommunikationsmanagement*: zur Sicherstellung der Verteilung wichtiger Informationen innerhalb vorgegebener Zeit an Entscheidungsträger
- *Qualitätsmanagement*: durch Verfahren wie Audit Trail, Validierungsfunktionen, Spezifikations-Libraries und graphische Wertedarstellung
- *Sicherheit*: Definieren von Zugriffsrechten, Paßworten, benutzerspezifischen Menüs und verschiedenen Zugriffsebenen
- *Elektronisches Laborbuch*: automatische Übertragung von erfassten Daten in ein elektronisches Laborbuch

LIMS können die Produktivität durch das Automatisieren beispielsweise folgender Laborprozesse steigern:

- Probenanmeldung / Probenlogin von Remote-Systemen
- Gerätewartungs- und Kalibrierungs-Logging
- Benachrichtigung, wenn ein Gerät out-of-calibration ist
- Zusammenführung von Ergebnissen verschiedener Tests in einen einzigen Report
- Reportgenerierung
- Dokumentieren, daß die korrekten Prozesse für die Probe abgearbeitet wurden

Die Entwicklung von LIMS wird auszugsweise in folgender Tabelle zusammengefaßt:

Jahr	Entwicklung
bis 1982	Verwendung von Laborbüchern und handgeschriebenen Reports zum Aufzeichnen der Informationen. Rudimentäre Ansätze für die Verwendung von Informationssystemen. Proprietäre Automatisierung von Laborgeräten durch den Gerätehersteller.
1982	Erste Generation der LIMS: Einführung des ersten kommerziellen LIMS. Sammeln von Labordaten auf einem zentralen Minicomputer, erste automatisierte Reporterstellung
1988	Zweite Generation der LIMS: Nutzen von relationalen Datenbanken auf einem zentralen Computer
1991	Dritte Generation der LIMS: Einführung von Client/Server-Konzepten; die Datenverarbeitung wird mit Laborcomputern und einem zentralen Datenbankserver realisiert
1995	Vierte Generation der LIMS: Ressourcensharing und Datenverarbeitung; Workload-Balan-

	cing zwischen den Servern und Clients
1996	Einführung von Web-enabled LIMS und drahtloser Kommunikation
1997	Integration elektronischer Unterschriften
1998	Integration von globaler Positionsbestimmung, um den Standort von Proben bei ihrer Entnahme zu dokumentieren (GPS)

Tabelle 4: Auszugsweise Übersicht über die Historie der Entwicklung von LIMS, angelehnt an [Gibbon 1996]. Der Fortschritt von Informationstechnologie und allgemeinen Architekturkonzepten beeinflusste stets die Entwicklung von Laborinformations-Managementsystemen.

3.4 Laborautomatisierungssoftware

Bei den in Kapitel 3.3 vorgestellten klassischen Automatisierungsansätzen agieren mehrere (eventuell teils konsolidierte[81]) Laborgeräte gleichzeitig bzw. interagieren miteinander[82]. Eine präzise[83] und robuste[84] Interaktion ist besonders bei allen Transport- und Übergabevorgängen von Proben zwischen den beteiligten Geräten erforderlich [Cahn 2007]. Die Koordination der einzelnen Geräte erfolgt im allgemeinen durch eine *Prozeß-Kontroll-Software* (*System Controller*), die das Verhalten des gesamten Systems bestimmt, indem sie die Ausführung mehrerer simultaner Bearbeitungsschritte (*Tasks* in der Terminologie von Laborautomatisierungssoftware, *Aktivitäten* in der Terminologie von Workflow-Managementsystemen) steuert (pro Gerät jeweils ein Task pro Zeitpunkt, beispielsweise für das Senden von Gerätekommandos oder für das Warten auf Geräteantwort). Nach [Berman 2007] werden zu diesem Zweck am häufigsten *statische Scheduler* verwendet. Aber auch andere Konzepte wie *dynamische Scheduler* oder *kooperatives*[85] *Multitasking* finden Anwendung. Die grundsätzliche Steuerbarkeit der jeweiligen Laborgeräte durch einen System Controller setzt eine entsprechende Integration des jeweiligen Geräts oder Moduls im automatisierten System voraus. Man kann bei der Geräteintegration im Wesentlichen zwischen dem traditionellen *monolithi-*

[81] Siehe Definition in Abschnitt 3.3.2.2

[82] Je höher die Anzahl der Geräte ist, desto schwieriger ist ihre präzise Koordination (vgl. [Cahn 2007]), beispielsweise bei der Übergabe von Proben zwischen verschiedenen Geräten.

[83] Hohe Präzision ist hier im Sinne mechanischer Akkuratheit und hinsichtlich exakten Timings bei der Interaktion der Geräte und Roboter zu verstehen.

[84] Hier ist im Besonderen das Vermeiden sogenannter *Deadlocks* (Betriebszustände eines System Controllers, die ohne eine Intervention des Benutzers nicht aufgelöst werden können) gemeint. Beispiel: eine Probe an Gerät A muß zu Gerät B transportiert werden, während eine Probe an Gerät B zu Gerät A transportiert werden muß. Wenn kein Zwischenlager existiert, liegt hier ein Deadlock vor.

[85] Ein spezieller Ansatz für die Implementierung kooperativen Multitaskings ist in [Cahn 2007] beschrieben. Es werden dabei sogenannte *Generatoren* (*wiederaufnehmbare Funktionen*, *resumeable functions*) verwendet, die bei Unterbrechung der Ausführung ihrer Instanzen intermediäre Werte beibehalten, die bei Wiederaufnahme der Ausführung ihrer Instanzen wieder zur Verfügung stehen.

schen Ansatz und der Verwendung sogenannter *Instrument Integration Frameworks (*oder *Device Integration Frameworks)* unterscheiden. System Controller und Geräteintegration leisten die eigentliche Automatisierung in Laborautomatisierungssystemen. Im Folgenden werden typische Prinzipien erläutert und Implementierungsansätze aus der Literatur beschrieben.

3.4.1 System Controller

Die simultane Ausführung mehrerer Tasks (*Multitasking*) ist unabhängig von Laborautomatisierung ein wesentlicher Aspekt der Informatik und speziell Gegenstand der Betriebssystemtheorie[86]. Da ein Mikroprozessor zu jedem Zeitpunkt höchstens eine Instruktion eines einzigen Tasks ausführen kann, muß der Eindruck von Multitasking durch sehr schnelles Umschalten zwischen den einzelnen Tasks erzeugt werden. Während des Zeitintervalls, in dem einem bestimmten Task benötigte Ressourcen wie Rechenzeit, Arbeitsspeicher oder Zugriff auf Peripheriegeräte zugewiesen sind, werden nacheinander eine oder mehrere Instruktionen dieses Tasks abgearbeitet. Die Aufgabe des Umschaltens und Zuweisens von Ressourcen wird von einem im jeweiligen Betriebssystem integrierten *Scheduler* erfüllt. Dieser ist für die Allokierung von Ressourcen und die zeitliche Planung von Tasks zuständig. Scheduler unterscheiden sich voneinander hinsichtlich ihrer Arbeitsweise, die von dem zugrundeliegenden Algorithmus definiert wird. Man unterscheidet *unterbrechende* (*präemptive*) und *nicht unterbrechende* (*kooperative*) Scheduling-Verfahren. Auf Betriebssystemebene ist ein Prozeß als Instanz eines Tasks (beispielsweise eines Anwendungsprogramms) definiert und wird vom Betriebssystem verwaltet. Im Allgemeinen laufen mehrere simultane Tasks auf einem Computer als jeweils eigene Prozesse ab. Kooperative Scheduler teilen einem Prozeß Ressourcen zu und warten auf das Freigeben der Ressourcen durch den Prozeß, so daß die Ressourcen einem anderen Prozeß zugewiesen werden können. Im Gegensatz dazu allokieren präemptive Scheduler die Ressourcen anhand bestimmter Kriterien und unterbrechen den jeweils aktiven Prozeß nach Ablauf der zugeteilten Zeit, unabhängig vom Abarbeitungsfortschritt des Prozesses. Es existiert eine Vielzahl unterschiedlicher Scheduling-Algorithmen für die unterschiedlichsten Anforderungen. Ein typisches Beispiel für präemptives Scheduling ist das ‚Round-Robin'-Verfahren, bei dem jedem Task eine identische Zeitscheibe zugewiesen wird, nach deren Ablauf das Task unterbrochen wird und wieder ans Ende einer zirkulären Task-Warteschlange gesetzt wird. Die meisten Betriebssysteme verwenden präemptive Schedulingalgorithmen. Im Allgemeinen ist zwischen einzelnen Tasks auf Betriebssystemebene keine Synchronisierung in der Form erforderlich[87], wie sie im Bereich automatisierter Systeme benötigt

---

[86] siehe beispielsweise [Tanenbaum 1995].

[87] Auf Betriebssystemebene stehen für die Koordination und Kommunikation zwischen einzelnen Prozessen (*inter-process communication*) nur diejenigen Möglichkeiten zur Verfügung, die das jeweilige Betriebssystem

wird, und es läßt sich nicht steuern, zu welchem Zeitpunkt ein aktives Task durch den Betriebssystem-Scheduler unterbrochen wird. Wie oben beschrieben, ist die genaue zeitliche Koordination zwischen einzelnen Tasks in automatisierten Laborsystemen für ihre präzise Interaktion jedoch unbedingt notwendig.

Würde man -rein hypothetisch betrachtet- den präemptiven Scheduler eines Betriebssystems zum Umschalten zwischen den einzelnen Tasks eines automatisierten Systems verwenden wollen (d.h. jedes Automatisierungstask direkt als eigenen Prozeß auf dem Betriebssystem ablaufen lassen) und die Synchronisierung dieser einzelnen Tasks beispielsweise durch eine gemeinsam genutzte Datei zu realisieren versuchen, könnten durch ungünstiges Unterbrechen der Tasks bestimmte Zustände eintreten, in denen die resultierende Reihenfolge von Instruktionen zu fehlerhaften Ergebnissen bzw. zum Systemausfall führt. Solche Zustände, die vom Timing bestimmter Instruktionen abhängen, werden als *race conditions*[88] bezeichnet. Ein Beispiel zur Veranschaulichung: zwei Titerplatten sollen durch dasselbe Gerät analysiert werden, das jedoch nur Kapazität für eine Titerplatte pro Zeitpunkt hat. Zwei Transportroboter unter Kontrolle jeweils eines separaten Tasks auf Betriebssystemebene wären in der Lage, jeweils eine Titerplatte zu dem Analysegerät zu bewegen. Eine hypothetische Instruktionsreihenfolge, entstehend durch präemptives Scheduling, ist in folgender Tabelle aufgeführt:

Zeitpunkt	Task 1 (für Roboter 1)	Task 2 (für Roboter 2)	Status
1	Read_instrument_status()		ready
2		Read_instrument_status()	ready
3	Set_instrument_busy()		busy
4		Set_instrument_busy()	busy
5	Move_sample_to_instrument()		busy
6		Move_sample_to_instrument()	Crash

Tabelle 5: Eine exemplarische Instruktionsreihenfolge, die zu *race conditions* führt. Ursache ist ungünstiges präemptives Scheduling der beiden Tasks (Task 1 steuert Roboter 1, Task 2 steuert Roboter 2). Beide Tasks werden zu ungünstigen Zeitpunkten unterbrochen, so dass ihre Instruktionen nicht wie erforderlich jeweils direkt nacheinander abgearbeitet werden. Stattdessen werden die Instruktionen der beiden Tasks abwechselnd ausgeführt, so dass der dem jeweiligen Task bekannte Allokierungs-Status des Analysegeräts vom tatsächlichen Allokierungs-Status abweicht. In der Konsequenz beginnen beide Tasks den Transport jeweils einer Titerplatte zum Analysegerät. Die Folge ist eine Kollision. Im Bereich der Laborautomatisierung muß daher das Umschalten zwischen einzelnen anstehenden Tasks gezielt beeinflusst werden können.

Es ist im Bereich der Laborautomatisierung also unbedingt erforderlich, das Umschalten der einzelnen Tasks gezielt beeinflussen und koordinieren zu können, aber auch die einzelnen

---

dafür vorsieht. Allen Betriebssystemen gemeinsam ist die Möglichkeit Datei-basierter Kommunikation und Synchronisierung.

[88] Solche Artefakte entstehen durch den Wettbewerb mehrerer Tasks um Allokierung von gemeinsam genutzten Ressourcen. Im Bereich der Laborautomatisierung sind die Ressourcen, um die zwischen den einzelnen Tasks ein Wettbewerb besteht, die Geräte und Module des automatisierten Systems.

Tasks miteinander synchronisieren zu können. Zu diesem Zweck werden geeignete Scheduler implementiert, die auf dem Betriebssystem aufsetzen[89] und die erforderlichen Möglichkeiten zur Synchronisierung und Koordination der einzelnen Automatisierungstasks, aber auch zur gezielten und steuerbaren Allokierung von Ressourcen bieten.

Im Zusammenhang mit automatisierten Laborprozessen sind die zu verwaltenden und zu allokierenden Ressourcen die Geräte und Module eines Laborautomatisierungssystems. Sie müssen den verschiedenen Tasks eines Protokolls zugewiesen werden. Scheduling im Kontext der Laborautomatisierung ist nach [Chen 1998] der Vorgang, verschiedene Alternativen zu berechnen und eine optimale Lösung auszuwählen, um jedem Task die benötigten Ressourcen und Zeitdauern zuzuweisen[90]. Dabei sind jeweils zeitliche Restriktionen der einzelnen Aktivitäten und die Kapazitäten der vorhandenen Ressourcen zu berücksichtigen. Scheduling ist somit ein Optimierungsprozeß, bei dem eine limitierte Anzahl von Geräten, beispielsweise Roboter oder andere Laborgeräte, für parallele und sequentielle Abarbeitung mehrerer Tasks allokiert werden. Für diese Zwecke optimierte Scheduler übernehmen die Zuteilung der Geräte an die verschiedenen Tasks und die Koordination ihrer Ausführung[91]. Man unterscheidet einerseits zwischen *statischen* und *dynamischen* Schedulern, und andererseits zwischen *single-threaded* und *multi-threaded* Schedulern. Im folgenden Abschnitt wird zunächst das Konzept des *Threadings* erläutert.

3.4.1.1 Threading

Auf Betriebssystemebene ist ein Prozeß als Instanz einer Applikation definiert und wird vom Betriebssystem verwaltet. Im Allgemeinen laufen mehrere simultane Applikationen auf einem Computer als jeweils eigene Prozesse ab. Es ist aber auch (abhängig von der verwendeten Programmiersprache, in der eine Applikation erstellt wird) möglich, mehrere kooperierende Tasks (beispielsweise ein Task für das Abfragen des Status eines Laborgerätes, ein Task

---

[89] Diese Scheduler werden als eigene Prozesse ausgeführt und laufen auf einem höheren Level als der Scheduler des Betriebssystems.

[90] Eine weniger flexible Möglichkeit zur Kontrolle automatisierter Systeme ist die Verwendung monolithischer Kontrollprogramme, die eine statische, vordefinierte Reihenfolge und Geräteallokierung vorgeben. Die Berechnung einer Reihenfolge durch einen Scheduler findet dabei nicht statt. Ab entsprechender Systemkomplexität ist ein solcher Ansatz jedoch ungeeignet, ebenfalls bei Existenz mehrerer Workflows, die vom System abgearbeitet werden sollen.

[91] In manchen Publikationen (beispielsweise [Schäfer 2004]) wird zwischen planender Komponente (Scheduler) und ausführender Komponente (*Exekutor*) unterschieden. Als Aufgaben eines Exekutors werden dabei das Senden von Gerätekommandos, das Abfragen von Timings und das Übertragen anfallender Werte an ein LIMS genannt. In der vorliegenden Arbeit wird auf diese Unterteilung verzichtet. Beide Aufgabenbereiche werden hier dem Scheduler zugeordnet.

zum Senden von Instruktionen an das Laborgerät, ein Task zum Auswerten von Daten) in einer einzigen Applikation zusammenzufassen und innerhalb eines einzigen Prozesses auszuführen. In solchen Fällen werden die einzelnen Tasks als separate *Threads* (oder *Threads of execution*) eines Prozesses bezeichnet. Threads stellen eine Möglichkeit dar, eine Applikation in mehrere (pseudo-)simultan ablaufende Stränge zu unterteilen (siehe unter anderem [Liberty 2002]). Einzelne Threads eines Prozesses können beispielsweise sehr schnell auf zeitkritische Ereignisse reagieren, während andere Threads beispielsweise langwierige Berechnungen durchführen. Dieses *Multithreading* ermöglicht einen effizienten Datenaustausch zwischen den verschiedenen Strängen und basiert auf der gemeinsamen Verwendung eines Speicherbereichs durch verschiedene Threads innerhalb eines Prozesses. Multithreading wird ähnlich wie Multitasking durch Umschalten zwischen den verschiedenen Threads erreicht. Eine Synchronisierung von Threads desselben Prozesses ist möglich, beispielsweise durch Sperren bestimmter Codeabschnitte eines Threads (*kritische Abschnitte*, *Locks*) [Cahn 2007], während deren Ausführung das Thread nicht unterbrochen werden darf[92].

Man unterscheidet beim Threading zwischen *kernel threads* und *user threads*[93]. Kernel threads werden durch den Scheduler des Betriebssystems verwaltet und analog zu normalen Prozessen gemäß dem Scheduling-Algorithmus des Betriebssystems behandelt. Das Scheduling von user threads erfolgt hingegen kooperativ durch Verwendung von Timern, Signalen oder anderen Methoden, die es user threads ermöglichen, ihre eigene Ausführung zu unterbrechen und anderen Threads die Ausführung zu erlauben. Unter Verwendung mehrerer user threads in einer einzigen Applikation lassen sich die auszuführenden Tasks einer Präparation vollständig vom Scheduler des Betriebssystems entkoppeln, in einem einzigen Prozeß organisieren und anhand eigener Scheduling-Algorithmen und mit Hilfe von Kommunikation zwischen den Threads gezielt steuern und koordinieren. Multithreading-fähige System Controller können für jedes Laborgerät jeweils spezifische Threads zur Kommunikation und Statusbestimmung nutzen [Berman 2007]. Single-threaded System Controller hingegen können lediglich ein einziges Thread für das Scheduling und die Kommunikation mit den Geräten verwenden.

Die beschriebenen Aspekte sind hinsichtlich statischer und dynamischer Scheduler im Kontext von Laborautomatisierung von Bedeutung.

---

[92] Unterbrechen ist hier im Sinn des Umschaltens zu einem anderen Thread desselben Prozesses gemeint.

[93] User threads werden auch als *fiber* bezeichnet.

3.4.1.2 Statische und dynamische Scheduler

Statische Scheduler berechnen die geeignete Allokierung der Laborgeräte und die Zuordnung der einzelnen Tasks, bevor das System gestartet wird. Ergebnis der Berechnung ist ein fester Arbeitsplan, der zur Ausführungszeit abgearbeitet wird [Schäfer 2004], [Cahn 2007], [Berman 2007]. Um einen optimierten Ablaufplan berechnen zu können, müssen detaillierte Informationen über die einzelnen Geräteoperationen (beispielsweise benötigte Zeit, relevante Deadlines, Prozesskosten) vorhanden sein. Auf Basis dieser Informationen ordnen statische Scheduler mit Hilfe heuristischer Strategien jeder Aktivität eines Protokolls ein Zeitfenster eines geeigneten Laborgerätes zu und berechnen auf diese Weise eine lineare Folge fest vorgegebener Ausführungszeitpunkte. Folgende Abbildung zeigt einen Arbeitsplan als Ergebnis statischen Schedulings:

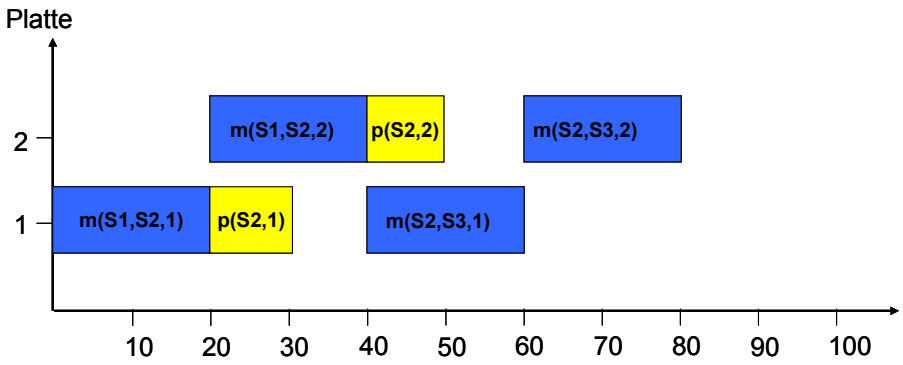

Abbildung 21: beispielhafter Arbeitsplan als Ergebnis statischen Schedulings, angeleht an eine Darstellung aus [Rodziewicz 2004]. Zwei Titerplatten sollen mit Hilfe eines Roboterarms innerhalb eines Systems transportiert und von mehreren Geräten bearbeitet werden. Die Notation ist wie folgt: Si:= Gerät i, m(Si, Sj, n):= benötigte Zeit, um Platte n zwischen Gerät i und Gerät j zu transportieren, p(Si, n):= Bearbeitungszeit an Gerät i für Platte n. Konkret wird hier dargestellt, daß zum Zeitpunkt t=0s der Transportvorgang von Titerplatte 1 von Gerät 1 nach Gerät 2 beginnen und nach t=20s abgeschlossen sein soll. Dann soll die eigentliche Bearbeitung der Titerplatte 1 an Gerät 2 beginnen, die mit einer Dauer von 10s eingeplant ist. Nachdem bereits nach t= 20s der Transportvorgang für Platte 1 abgeschlossen sein soll und der Roboterarm als frei angenommen wird, soll der Transport von Titerplatte 2 zu Gerät 2 beginnen. Dieser ist laut Arbeitsplan zum Zeitpunkt t=40s abgeschlossen, so daß dann die Bearbeitung von Titerplatte 2 beginnen soll. Der Roboterarm kann laut Arbeitsplan zu diesem Zeitpunkt mit dem Transport von Titerplatte 1 zu Gerät 3 beginnen, der zum Zeitpunkt t = 60s beendet sein soll, so daß dann Titerplatte 2 zu Gerät 3 transportiert werden soll.

Bei der Ausführung wird nur ein einziges Thread benötigt, um die einzelnen Geräte nacheinander zu steuern [Berman 2007]. Zur Ausführungszeit kann ein berechneter Arbeitsplan nicht verändert werden [Schäfer 2004]. Dies führt zu Schwierigkeiten bei der Handhabung auftretender Fehler zur Laufzeit und beeinträchtigt von vornherein die Robustheit eines Systems. Abweichungen zwischen geschätzter Zeit und tatsächlich benötigter Zeit für einen Bearbeitungsschritt müssen vermieden werden, um Timing-Probleme bei der Interaktion der Geräte zu verhindern [Rodziewicz 2004]. Außerdem kann die Ausführung eines Arbeitsplans nicht vom aktuellen Status von Geräten oder dem Bearbeitungsfortschritt beeinflußt werden.

Dies erlaubt beispielsweise keine *event-basierte* Durchsatzsteigerung, aber vor allem auch keine weitere Bearbeitung von Proben im Fehlerfall, der stattdessen den Ausfall des gesamten Systems nach sich zieht [Berman 2007].

Im Gegensatz zu statischem Scheduling werden beim dynamischen (*event-driven*) Scheduling die Allokierung der Laborgeräte und die Zuordnung der einzelnen Tasks erst zur Laufzeit des Systems entschieden. Dynamisches Scheduling basiert auf den Parametern und Randbedingungen (*Constraints*), die mit den jeweiligen Tasks assoziiert sind. Entscheidungen über die Abfolge der anstehenden Bearbeitungsschritte erfolgen kontinuierlich, auch unter Berücksichtigung auftretender Events, beispielsweise Statusmeldungen, Fehlerzustände oder dem aktuellen Bearbeitungsfortschritt. Der ursprüngliche Arbeitsplan wird zur Laufzeit entsprechend dynamisch angepaßt [Schäfer 2004]. Die folgende Abbildung veranschaulicht (1) die Interaktion eines dynamischen Schedulers mit dem Status von Laborgeräten und dem Bearbeitungsfortschritt zur Laufzeit und (2) die Rückkopplung auf den verbleibenden Arbeitsplan:

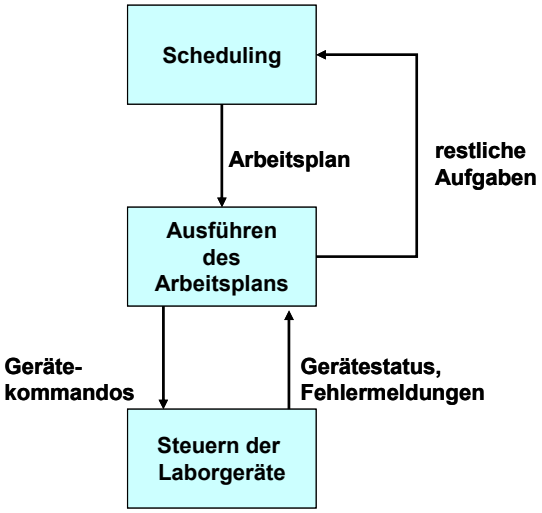

Abbildung 22: Schematische Darstellung event-basierten Schedulings mit Interaktion zwischen Events und dem Anpassen des restlichen Workflows, angeleht an [Schäfer 2004]. In Abhängigkeit vom Status der Geräte oder eventuellen Fehlern wird ein neuer Arbeitsplan für die noch verbliebenen Aufgaben dynamisch erstellt.

Um einerseits Geräte zu steuern, andererseits auf Zustände und Events sofort reagieren zu können (beispielsweise einen robotisierten Transport auslösen, wenn die Bearbeitung einer Probe durch ein Gerät abgeschlossen ist), benötigen dynamische Scheduler grundsätzlich mehrere Threads. Multithreading wird bei event-basierten System Controllern außerdem verwendet, um höhere Robustheit des Systems und höheren Durchsatz zu erzielen. Im Fall eines Geräteausfalls arbeiten die Threads der anderen Geräte weiter, und der Arbeitsplan wird gegebenenfalls dahingehend angepaßt, daß die Aufgaben des defekten Geräts von einem anderen Gerät übernommen werden. Die Entscheidung für event-basiertes Scheduling erfordert die Verwendung geeigneter Programmiersprachen, die Multithreading unterstützen. Berman (vgl. [Berman 2007]) beschreibt in diesem Kontext C++ als eine sehr gut geeignete Programmiersprache, während er VisualBasic wegen fehlenden Multithreadings als ungeeignet klassifi-

ziert. Nachfolgende Übersicht vergleicht die wesentlichen Merkmale statischer und dynamischer (event-driven) Scheduler miteinander:

	Statische Scheduler	**dynamische Scheduler**
Im Fehlerfall	Alle Vorgänge werden gestoppt.	Vorgänge können gestoppt werden oder weiterlaufen, je nach Benutzerentscheidung.
Gerätepooling	Kann den Probendurchsatz erhöhen, erzeugt aber keine Redundanz.	Kann den Probendurchsatz erhöhen und Redundanz erzeugen.
Durchsatz	Keine Maximierung möglich, solange die exakten Dauern der einzelnen Tasks nicht genau bekannt sind. Da der Scheduler nicht weiß, wann genau Tasks abgeschlossen sind, können sie nicht unmittelbar weiterbearbeitet werden, sondern erst zu dem im Arbeitsplan berechneten Zeitpunkt für den nächsten Bearbeitungsschritt.	Kann maximiert werden, da die Weiterverarbeitung sofort nach Beenden eines Tasks beginnen kann.
Startdauer	Langsam, denn der Scheduling-Algorithmus muß zunächst den optimalen statischen Arbeitsplan berechnen, bevor die Ausführung beginnen kann.	Sehr schnell, da vorab keine Arbeitsplan-Berechnung erfolgen muß.
Zeitliche Vorhersagbarkeit	Exakt wegen vorberechneten Zeitplans	Nicht exakt vorhersagbar
Deadlock-Zustände	Können nicht auftreten, da lineare Abarbeitung des Zeitplans in einem einzigen Thread erfolgt.	Können auftreten wegen der Existenz mehrerer Threads.

Tabelle 6: Vergleich statischer und dynamischer Scheduler (angelehnt an [Berman 2007]). Während statische Scheduler systembedingt eine präzise zeitliche Vorhersagbarkeit erlauben, können dynamische Scheduler flexibel auf eintretende Ereignisse reagieren und den Arbeitsplan anpassen. Daraus resultiert eine höhere Ausfallsicherheit und Redundanz.

Die gute zeitliche Vorhersagbarkeit des Systemverhaltens (fehlerfreier Betrieb und akkurates Timing vorausgesetzt) macht statisches Scheduling zu einer weit verbreiteten Lösung für viele Automatisierungssysteme. Dynamisches Scheduling ist jedoch die bessere Alternative, wenn Echtzeit-Automation in einem dynamischen Umfeld implementiert werden soll oder Robustheit und hoher Durchsatz benötigt werden. Dynamisches Scheduling benötigt für das kontinuierliche und iterative Erstellen und Korrigieren von Ablaufplänen in Echtzeit allerdings höhere Rechenkapaziäten. Die Wahl zwischen statischem und dynamischem Scheduling ist somit ein Kompromiß zwischen zeitlicher Vorhersagbarkeit, Komplexität der Tasks, Effizienz des Scheduling-Algorithmus, Fehlertoleranz, erforderlichem Throughput der automatisierten Bearbeitung und Einhaltung spezifizierter Constraints (beispielsweise Deadlines).

Ein weiterer Ansatz für präzise Laborautomatisierung wird in der Verwendung von *Echtzeit-Betriebssystemen* (*real time operating systems* (*RTOS*)) gesehen. RTOS garantieren

jeweils eine definierte Anfangszeit für die anstehenen Tasks, eine definierte Ausführungsdauer und die Einhaltung zeitlicher Constraints. Dies erfolgt durch spezialisierte Scheduling-Algorithmen. Trotz dieser Vorteile werden sie wegen der benötigten spezifischen Programmiertools, des spezifischen Wissens für die Anwendungsprogrammierung und der erforderlichen System-Anforderungen häufig als unpraktikabel eingestuft [Cahn 2007]. Eine Übersicht über Echtzeit-Betriebssysteme und eine Motivation ihres Einsatzes im Bereich der Laborautomatisierung wird in [Cedeno 2007] gegeben.

3.4.2 Integration von Laborgeräten in Laborautomatisierungssysteme

3.4.2.1 Traditioneller Ansatz

Gemäß dem traditionellen Ansatz (zusammenfassend beschrieben in [Thakur 1998] und [McIntosh 2003]) werden Laborautomatisierungssysteme genau für eine einzige oder für wenige exakt definierte Anwendungen konstruiert. Zu integrierende Geräte und sonstige Hardware (beispielsweise Sensoren) werden hinsichtlich ihrer Eignung für diesen begrenzten Anwendungsbereich ausgewählt, und System Controller und erforderliche Software auf Geräteebene werden speziell für die jeweilige Anwendung implementiert. Die Befehle zur Ansteuerung der Geräte sind im System Controller hardcodiert, so dass dieser die konkreten Gerätekommandos und deren Parameter verwendet, um Geräte direkt anzusteuern. Folgende Abbildung soll diesen Aufbau illustrieren:

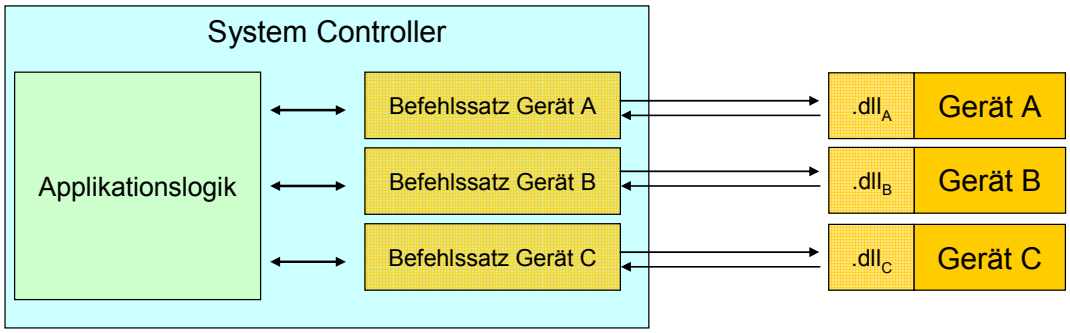

Abbildung 23: schematische Illustration traditioneller Geräteansteuerung in Laborautomatisierungssystemen. Die wesentlichen Bestandteile sind der System Controller und Software auf Geräteebene, die in diesem Beispiel in Form gerätespezifischer .dlls ausgeführt ist. Im System Controller sind die konkreten Befehle zum Aufruf der Funktionen der .dlls hardcodiert. Die Geräteansteuerung (oder –abfrage) erfolgt über Befehlsaufrufe durch den System Controller an die .dll des entsprechenden Gerätes.

Solche *monolithischen* Architekturen sind bei ihrer Implementierung mit hohem Zeitaufwand verbunden, und andererseits machen sie Anpassungen und Erweiterungen am System schwierig [McIntosh 2003]. Konsequenz ist die weitgehende Unflexibilität solcher Systeme. Erweiterungen auf Geräteebene erfordern stets zusätzlich zur Implementierung der

benötigten Gerätesoftware auch das Redesign oder Reengineering des System Controllers[94]. Daher ist die spätere Anpassung eines solchen Systems, beispielsweise zur Integration eines neuen Gerätes, mit signifikantem Aufwand und entsprechenden Ausfallzeiten verbunden[95]. Wenn das System zusätzlich validiert ist, erfordert jede Änderung eine erneute Validierung. Eine Softwareeimplementierung, die beispielhaft für diesen traditionellen Ansatz ist, wird in [Graves 2000] im Zusammenhang mit der Entwicklung eines automatisierten Biorepositories für humanes genetisches Material beschrieben. Dieses Biorepository verwendet Robotik und automatisierte Laborgeräte zum Analysieren, Archivieren und Entnehmen von aufbereiteten humanen DNA- bzw. RNA-Proben, die für wissenschaftliche Studien benötigt werden. Es wurde aus mehreren Serienprodukten und aus eigenentwickelten Komponenten aufgebaut. Neben einem anthropomorphen Roboterarm (CRS Robotics T255) wurden folgende Geräte integriert:

- 'TECAN Genesis RSP150' liquid handling station
- 'TECAN SPECTRAFLUOR PLUS' microplate scanner
- 'TECAN MOL BANK' automated microplate storage unit
- 'ABGene ALPS-100' plate sealer
- 'Sartorius LP 820' balance
- 'Torrey Pines Scientific EchoTherm' chiller

Der System Controller[96] ist als *GUI-Applikation* implementiert und übernimmt zusätzlich zur Steuerung der integrierten Module und Geräte auch Aufgaben wie System-Monitoring und Kommunikation mit einem SQL-Server. Er kommuniziert mit verschiedenen gerätespezifischen Applikationen, beispielsweise mit der herstellerproprietären Applikation ‚CROSnt' des Roboterarms oder der herstellerproprietären Applikation ‚Gemini' der Tecan Genesis Workstation. Für die Steuerung dieser beiden Applikationen wurde jeweils eine zusätzliche Applikation als Schnittstelle zwischen Geräteapplikation und System Controller implementiert. Andere Geräte des Biorepositories werden durch unmittelbar im System Control-

---

[94] Dies ist unter anderem auf teils sehr unterschiedliche, herstellerspezifische Kommunikationsprotokolle der verwendeten Laborgeräte zurückzuführen. Es gibt zwar diesbezügliche Standards, beispielsweise den in [Staab 1999] beschriebenen *Laboratory Equipment Control Interface Standard* (*LECIS*), aber tendenziell werden solche Standards lediglich als Vorlage für die Eigenentwicklung von Protokollen verwendet (vgl. [McIntosh 2004], [McIntosh 2003]). Eine konsequente Umsetzung von LECIS zeichnet sich nicht ab, stattdessen manifestiert sich eine geringe Akzeptanz dieses Standards am Markt [McIntosh 2003], und eine Referenz-Implementierung von LECIS fehlt. Die Anbindung von Geräten wird herstellerseitig nach wie vor meist durch *dynamic link libraries* (*.dll*) oder durch *ActiveX-Libraries* unterstützt.

[95] Diese Problematik besteht beim Einsatz monolithischer Laborautomatisierungssoftware sowohl für TLA als auch für automatische Labormodule.

[96] in [Graves 2000] als Prozeß-Kontroll-Logik bzw. als Betriebssystem bezeichnet

ler integrierte Gerätekommandos und –parameter gesteuert, beispielsweise eine Waage oder das automatische Tieftemperatur-Lagersystem. Der monolithische Charakter des Systems resultiert in den oben beschriebenen Problemen hinsichtlich Flexibilität und Erweiterbarkeit.

3.4.2.2 Device Integration Frameworks

Beim oben beschriebenen traditionellen Ansatz zur Geräteintegration stellen konkrete Gerätekommandos und ihre Parameter einen festen Bestandteil des System Controllers dar. Bei Änderung oder Erweiterung an der Geräteausstattung eines solchen Laborsystems muß stets auch der System Controller überarbeitet werden. Um dies zu vermeiden, zielen sogenannte *Device Integration Frameworks*[97] auf die vollständige Entflechtung von System Controller (*Applikationslevel*) und Software auf Geräteebene (*Gerätelevel*). Device Integration Frameworks sind auf Geräteanbindung an einen System Controller und auf Gerätesteuerung optimierte Sammlungen wiederverwendbarer Klassen und Komponenten, die von Applikationen im Bereich der Laborautomatisierung häufig benötigt werden[98]. Bei der Verwendung solcher Frameworks werden die konkreten, spezifischen Kommandos von Laborgeräten nicht im System Controller implementiert. Vielmehr wird die Integration und Steuerung von Laborgeräten durch die Komponenten des verwendeten Frameworks realisiert. Der System Controller interagiert über definierte Schnittellen mit dem Framework, das mit *peripheren Softwarekomponenten*[99] auf Gerätelevel kommuniziert. Der System Controller steuert somit Laborgeräte und Module mittelbar durch Einsatz einer Zwischenebene. Mit Hilfe dieser Zwischenebene ist es auch möglich, konkrete Befehle zum Event-Handling oder für das Kommunikationsmanagement aus dem System Controller auszulagern und stattdessen in Komponenten eines Frameworks zu implementieren. Folgende Ziele werden mit der Verwendung von Device Integration Frameworks im Zusammenhang mit Laborautomatisierung verfolgt:

- Flexibilität hinsichtlich der Geräteausstattung
- Erweiterbarkeit
- zeitnahe System-Rekonfigurierbarkeit
- Robustheit

---

[97] Ein eigenes Device Integration Framework wird im Zusammenhang mit der Implementierung des GHRC-Prototyps in Kapitel 5.5 beschrieben.

[98] Allgemein bezeichnet der Begriff *Framework* eine Sammlung wiederverwendbarer Klassen, die von Applikationen einer bestimmten Domäne häufig benötigte Funktionalitäten bereitstellen. Ein Framework wird als Gerüst für die Entwicklung von Applikationen genutzt, stellt aber auch die Umgebung für die Ausführung der Applikationen zur Laufzeit dar.

[99] Als periphere Komponenten werden Software-Komponenten bezeichnet, die zur Steuerung der Funktionalitäten einzelner Peripherie- bzw. Laborgeräte dienen.

- Wiederverwendbarkeit von Softwarekomponenten

Um diese Ziele zu erreichen, ist trotz enger Interaktion der Komponenten eine lockere Kopplung, beispielsweise über schmale, wohldefinierte Schnittstellen, erforderlich. Dies erlaubt zudem die voneinander unabhängige Verbesserung einzelner Komponenten des Frameworks. Folgende Abbildung soll die Funktionsweise von Device Integration Frameworks veranschaulichen:

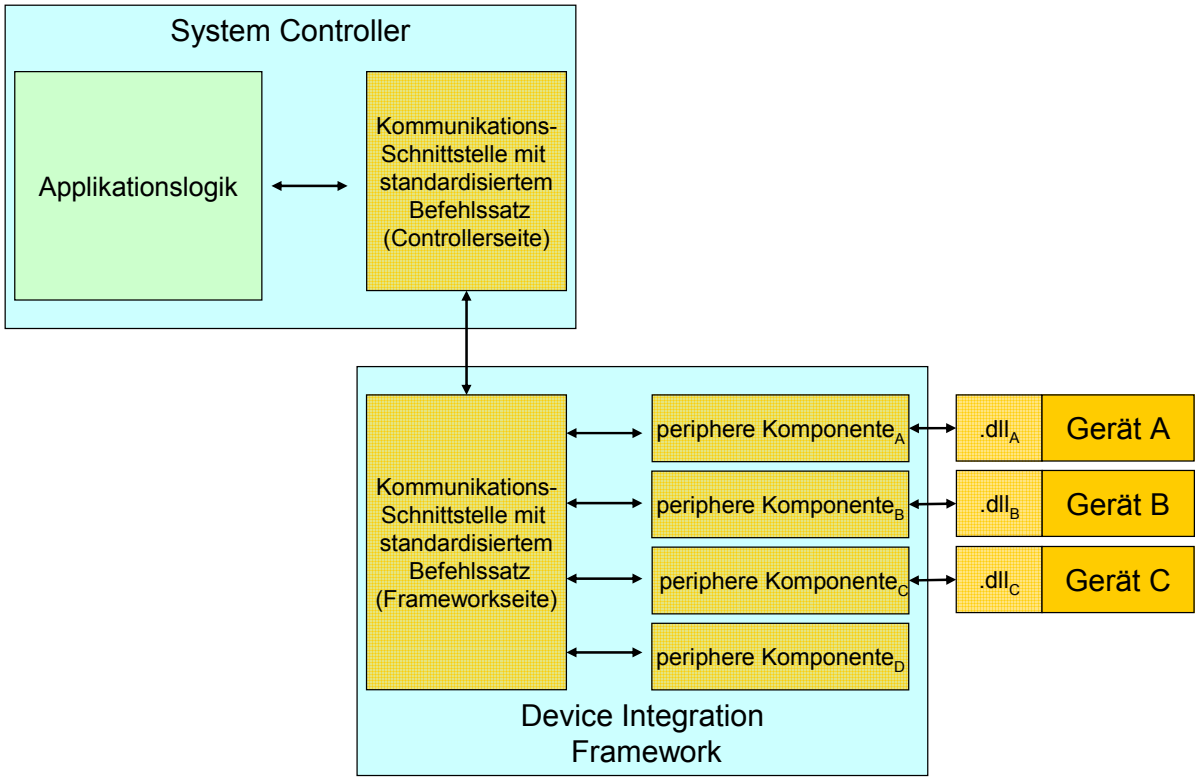

Abbildung 24: Illustration der Entkopplung von System Controller und Software auf Gerätelevel durch Verwendung eines Device Integration Frameworks. Die konkreten Befehle zum Aufruf der Funktionen gerätespezifischer .dlls sind hier nicht im System Controller hardcodiert, sondern in wieder verwendbaren, gerätespezifischen peripheren Softwarekomponenten des Frameworks. Der System Controller kommuniziert mit dem Framework über eine standardisierte Schnittstelle. Das Framework setzt die standardisierten Befehle des System Controllers durch Verwendung der entsprechenden peripheren Komponente in konkrete Funktionsaufrufe an die .dll des Gerätes um. Die Erweiterung eines Laborautomatisierungssysems um weitere Geräte erfordert nur die Erweiterung des Frameworks um die benötigten Komponenten. Eine Anpassung des System Controllers ist nicht erforderlich.

Mit der Verwendung von Device Integration Frameworks verschiebt sich der Fokus hin zur Entwicklung wiederverwendbarer Komponenten, die wiederholt für Aufbau oder Erweiterung automatisierter Systeme genutzt werden können. Im Gegensatz zu hoch spezialisierten monolithischen Systemen bilden die Komponenten von Device Integration Frameworks eine generische Menge von Funktionen ab. Sie stellen benötigte Standard-Funktionalitäten zur Verfügung und kommunizieren über definierte Schnittstellen untereinander und mit Software auf Applikationslevel. Als Beispiele für solche Komponenten nennt Berman (vgl. [Berman 2007]) Gerätetreiber, Benutzerschnittstellen für Laborgeräte oder Kommunikations-

protokolle. Die kontinuierliche Erweiterung eines solchen Frameworks um neue Komponenten führt letztlich zu einer umfangreichen Komponentensammlung, die zur Integration weiterer Laborgeräte oder dem Aufbau anderer Laborautomatisierungssysteme verwendet werden kann.

Es existieren in der Literatur zahlreiche verschiedene Ansätze für Device Integration Frameworks, die sich hinsichtlich der zugrundeliegenden Konzepte, der zu erfüllenden Anforderungen, der verwendeten Programmiersprachen oder Programmierumgebungen voneinander unterscheiden. Einige Beispiele sind in [Rodziewicz 2004], [Elliott 2007], [Thakur 1998], [Chen 1998] oder [Hung 2004] beschrieben. Die meisten Scheduler und Device Integration Frameworks sind in VisualBasic, C/C++ oder LabView implementiert [Rodziewicz 2004]. Berman (vgl. [Berman 2007]) nennt die Prinzipien Kapselung, Vererbung und Polymorphismus als Gründe, weshalb C++ für die Implementierung der Komponenten eines Device Integration Frameworks anderen Programmiersprachen vorzuziehen sei. Gemäß den Ausführungen in [Chen 1998] wird die Entwicklung eines vollständigen Frameworks für die komplexe Domäne der Laborautomatisierung wegen der unüberschaubaren potentiellen Anwendungen und verschiedenartigster Laborgeräte als schwierig angesehen.

Exemplarisch werden im Folgenden einige ausgewählte Anforderungen an ein Framework für die automatisierte Sequenzierung von Genomen aufgelistet, die in [McIntosh 2003] definiert werden:

- hoher Modularisierungsgrad des Frameworks
- standardisierte Geräteinterfaces
- Definition eines allgemeinen Kommando- und Instruktionsprotokolls für die Interaktion von Geräten über TCP/IP[100]
- *wrapper code* für die Integration herstellerspezifischer, existierender Gerätesoftware
- High-Level-Funktionen als zweckoptimierte Zusammenfassung einzelner Low-level-Routinen[101]

In [Rodziewicz 2004] wird eine Software-Suite zum Aufbau von Laborautomatisierungssystemen beschrieben. Diese enthält einen hybriden[102] Scheduler (‚Supra') als System Controller, ein Device Integration Framework (‚Genera') zur Geräteintegration, ein Simula-

[100] In Kapitel 5.5 wird mit der Implementierung des *virtuellen DCManagers* in Verbindung mit der beschriebenen Kommunikation zwischen *Commander* und Geräten eine ähnliche Anforderung realisiert.

[101] Solchen High-Level-Funktionen entsprechen die in Kapitel 5.5 im Zusammenhang mit der Implementierung des *DCMS* entwickelten *DCJobs*.

[102] Rodziewicz (vgl. [Rodziewicz 2004]) bezeichnet den Scheduler ‚Supra' als hybrid, weil er statisches und dynamisches Scheduling unterstützt.

tionstool (‚SimView') und ein Web-enabled LIMS. ‚Genera' verwendet XML-basierte Gerätedefinitionen (bestehend aus Geräteeigenschaften und Gerätekommandos), aus denen gerätespezifische GUI-Komponenten generiert werden. Die Kommunikation zwischen dem Scheduler und den Komponenten des Frameworks basiert auf *Messages*. Diese Suite wurde für den Aufbau eines kundenindividuellen, automatisierten *ELISA*-Systems verwendet.

4 Konzeption der Arbeit

4.1 Beschreibung des Forschungskontextes

Die vorliegende Arbeit steht in direktem Zusammenhang und in unmittelbarer Wechselwirkung mit der Entwicklung einer zukunftsweisenden Technologieplattform für die vitale Langzeitlagerung von Zellen unter kryogenen Bedingungen. Kernelement dieser Technologieplattform sind Probenträger (*Substrate*) mit integriertem elektronischem Speicherchip[103]. [Durst 2003] war zuvor in demselben Umfeld entstanden und fokussierte auf Anforderungen an geeignete Probendatenbanken.

Motivation einer direkten Kopplung von biologischer Probe und nichtflüchtigem, elektronischem Speicherchip ist das Streben nach angemessen hoher Qualität im Biobanking, die durch dezentrale Datenhaltung bei den Proben selbst erreicht werden soll. Im Vordergrund stehen dabei eine zuverlässige Identifizierung von Proben, eine verwechslungssichere Wissensbewahrung und ein vereinfachter Wissensaustausch. Der Ansatz des Fraunhofer-IBMT unterscheidet sich von anderen[104] in der Literatur beschriebenen Ansätzen zu elektronischer Probenidentifikation (beispielsweise [Bettendorf 2005]) durch den Anspruch, ein umfangreiches Wissen dezentral bei der Probe zu speichern, das deutlich über eine reine Identifizierung hinausgeht und bisher ausschließlich in Probendatenbanken, Dokumentenmanagementsystemen oder papierbasierten Systemen organisiert werden konnte.

Dieser dezentrale Ansatz des Fraunhofer-IBMT zur Qualitätssicherung (*Quality Assurance, QA*) im Biobanking fokussierte anfangs ausschließlich auf den Umgang mit vorhandenem deklarativem Wissen, also auf die Aspekte der Wissensrepräsentation und Wissenskommunikation gemäß dem Prozeßmodell nach [Reinmann-Rothmeier 2000], nicht aber auf die Qualität des Wissens selbst. Gerade dieser Aspekt jedoch ist hinsichtlich des wissenschaftlichen Wertes von Probensammlungen von herausragender Bedeutung und erfordert eine Qualitätssicherung bereits während der Wissensgenerierung. Der in Kapitel 4.5 formulierte eigene Ansatz widmet sich dieser Problematik.

Simultan zur Entstehung der vorliegenden Arbeit[105] wurden prototypische Substrate mit integriertem Speichermedium bis zur Serienreife entwickelt und darauf basierend eine umfas-

---

[103] Elektronische Speicherchips müssen in diesem Zusammenhang den biophysikalisch benötigten kryogenen Temperaturen über lange Zeit standhalten.

[104] Motivation für elektronische Probenidentifizierung sind unter anderem Unzulänglichkeiten optischer Verfahren (beispielsweise handschriftlicher Kennzeichnungen).

[105] Die Entwicklung der Technologieplattform wird unter anderem von Aspekten des eigenen Ansatzes und daraus resultierenden Anforderungen beeinflußt.

sende elektronische Lagertank-Infrastruktur konzipiert, die mittlerweile erste Serienreife erreicht hat. In den Abschnitten 4.3 und 4.4 werden ausgewählte Substrattypen und verschiedene Komponenten der realisierten Technologieplattform insoweit beschrieben, als sie im Zusammenhang mit dem in Kapitel 4.5 beschriebenen eigenen Ansatz stehen und für die in Kapitel 5 beschriebenen Ergebnisse der vorliegenden Arbeit von Bedeutung sind.

4.2 Beschreibung des Projektkontextes

Die Technologieplattform des IBMT wie auch die vorliegende Arbeit sind außerdem in ein internationales Forschungsnetzwerk involviert, das von der *Bill & Melinda Gates Foundation* gefördert wird. Es handelt sich dabei um die *Collaboration for AIDS Vaccine Discovery* (*CAVD*), deren Ziel die Entwicklung eines Impfstoffes gegen HIV/AIDS ist. Die CAVD wurde im Juli 2006 gegründet, umfaßt 89 Institutionen aus 22 Ländern und unterstützt die *Global HIV/AIDS Vaccine Enterprise* (*GHAVE*)[106]. Dabei setzt sie auf kollaborative und vergleichende Forschung, die durch innovative Konzepte und neue Technologien unterstützt werden soll[107]. Motiviert wird dieser Ansatz durch die Hoffnung, mit Hilfe weltweit koordinierter Forschung und komparativen Impfstoff-Studien eine gemeinsame Wissensbasis aufbauen zu können, die die Entwicklung eines funktionierenden HIV-Impfstoffes ermöglicht[108]. In diesem Zusammenhang sind die technologischen Konzepte des IBMT zur Qualitätssicherung im Biobanking, zum Wissensmanagement durch dezentrale Speicherchips und der in Kapitel 4.5 beschriebene eigene Ansatz zur Qualitätssicherung bei der Wissensgenerierung von Bedeutung[109]. Das Fraunhofer-IBMT koordiniert ein internationales Konsortium zum Aufbau der zentralen HIV-Probenbank der CAVD. Dieses *Global HIV Specimen Research Cryorepository* (*GHRC*) soll der Lagerung von HIV-Proben und Reagenzien unter kontrollierten Bedingungen und ihrem Austausch mit Laboratorien und Konsortien innerhalb der CAVD dienen. Diese Biobank ist die erste Groß-Biobank weltweit zur Kryokonservierung HIV-relevanter Proben und Reagenzien und wird seit 2007 im Fraunhofer-IBMT in Sulzbach aufgebaut. Die angegliederten S3-Laboratorien arbeiten gemäß der *Good Clinical Laboratory Practice* (*GCLP*).

---

[106] Die Global HIV/AIDS Vaccine Enterprise (GHAVE) ist eine internationale Allianz, die 2004 zur Entwicklung eines HIV-Impfstoffes gegründet wurde. Ihr *scientific strategic plan* wurde 2005 veröffentlicht [GHAVE 2005] und hat als oberste Ziele die Entwicklung eines Impfstoffes und eine Standardisierung von Laboratorien.

[107] Speziell für Standardisierung und Wissensaustausch

[108] Dies erscheint aufgrund der hohen Mutationsrate des HI-Virus notwendig.

[109] Dies gilt besonders vor dem Hintergrund, dass es bislang keine Standards zum elektronischen Austausch von deklarativem Probenwissen, SOPs oder Labordokumentation zwischen den einzelnen Konsortien, Forschungszentren und Primary Sites gibt.

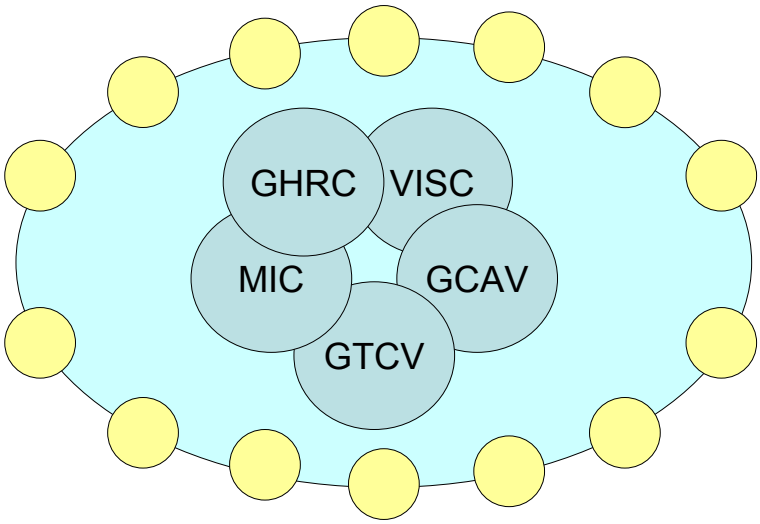

Abbildung 25: Schematische Darstellung der Collaboration for AIDS Vaccine Discovery (CAVD). Fünf zentrale Service-Institutionen unterstützen 14 Konsortien, deren Ziel die Entdeckung eines HIV-Impfstoffes ist. Insgesamt sind 89 Institutionen aus 22 Ländern in die CAVD involviert. Kooperative Forschung und komparative Studien sollen dem Aufbau einer gemeinsamen Wissensbasis dienen. Neben dem Global HIV Vaccine Research Cryorepository (GHRC), das die zentrale Biobank für die Lagerung und den Austausch von HIV-relevanten Proben und Reagenzien innerhalb der CAVD darstellt und entsprechendes deklaratives Wissen mit den Konsortien austauscht, unterstützen drei Labornetzwerke (Schwerpunkt: Evaluierung von Immunantworten auf potentielle Impfstoffe) und ein Statistikzentrum (VISC) die Impfstoffkonsortien. VISC tauscht mit den Labornetzwerken und den Impfstoffkonsortien deklaratives Wissen in Form von Ergebnissen und statistischen Daten aus. Neben dem Aufbau einer funktionalen HIV-Kryobank sind die Erforschung HIV-spezifisch optimierter Kryoprozeduren, die Entwicklung einer HIV-spezifisch optimierten Technologieplattform und die Laborstandardisierung weitere Aufgaben des GHRC-Konsortiums. Prozedurales Wissen in Form standardisierter Protokolle wird vom GHRC-Konsortium den übrigen Konsortien zur Verfügung gestellt.

Das GHRC-Konsortium besteht aus dem Fraunhofer-IBMT in St. Ingbert, der World Health Organisation (WHO) in Genf, dem National Institute for Biological Standards and Control (NIBSC) in London, dem San Raffaele Scientific Institute (DIBIT) in Mailand, der University of Lund, der Universität des Saarlandes, der University of Washington, dem National Institute of Health (NIH) und aus sogenannten *Primary Sites* in Ländern mit hoher HIV-Infektionsrate. Die Aufgabe der Primary Sites besteht im Sammeln von Proben und ihrer Aufarbeitung vor Ort und dem Austausch mit der GHRC-Biobank. Eine weitere Aufgabe des GHRC-Konsortiums ist das Erarbeiten von Laborstandards und biologischer Prozeduren Solche Standards (beispielsweise optimierte Verfahren zur Kryokonservierung oder SOPs für die Virusisolation) sind Voraussetzung für Vergleichbarkeit von Ergebnissen und somit für komparative Studien. Die Technologieplattform des IBMT wird im Rahmen dieses Projekt-Engagements mit Fokus auf ihren Einsatz innerhalb des GHRC-Konsortiums weiterentwickelt und angewandt. Die dabei entstehenden Technologien sollen zu einem späteren Zeitpunkt allen Konsortien der CAVD und GHAVE zur Verfügung gestellt werden.

4.3 Substrate mit elektronischem Speichermedium

Untersuchungen an kommerziell erhältlichen Flash-Speichermedien[110] auf Kryotoleranz (zusammengefaßt in [Ihmig 2006]) waren Ausgangspunkt für die Konstruktion erster prototypischer Substrate mit integriertem Compact-Flash-Speicher. Die Einbeziehung logistischer[111] wie auch biophysikalischer[112] Aspekte in das Design hat zusammen mit dem Compact-Flash-Formfaktor zunächst stapelbare, mehrere Probentöpchen enthaltende Substrate ergeben. Bei diesem Konzept wurde eine gemeinsame Nutzung des Speichermediums für alle aufgeschobenen Substrate und deren einzelne Wells vorgesehen. Dies spiegelt sich in der in Kapitel 5.2 beschriebenen ersten Datenstruktur für dezentrale Wissensbewahrung wider, die auf Basis der nachfolgend abgebildeten Substratstapel konzipiert wurde.

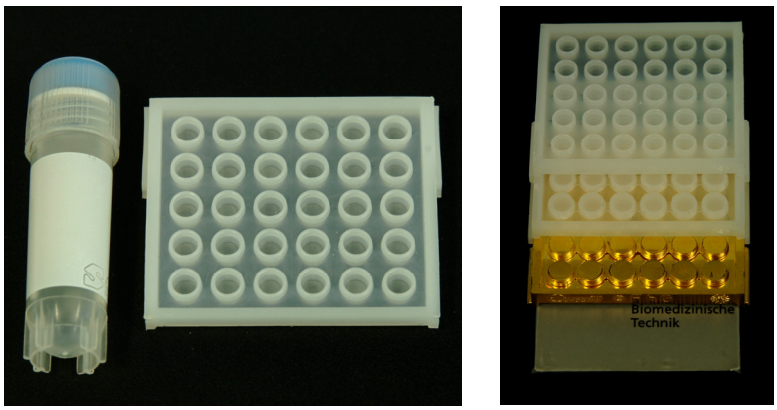

Abbildung 26: Links: Größenvergleich zwischen einem konventionellen Standard-Kryoröhrchen mit 2ml Volumen und einem IBMT-Multiwellsubstrat mit 30 Töpfchen zu jeweils 25µl. Rechts: Substratstapel aus jeweils zu Illustrationszwecken halb aufeinander geschobenen Multiwellsubstraten, halb aufgeschoben auf einen Compact-Flash-Speicher. Eines der abgebildeten Substrate ist zum Test auf spezifische Biokompatibilität mit einer Goldschicht besputtert.

Begleitende Entwicklungen zum Befüllen oder Verschließen der Substrate und für das vereinfachte Aufschieben eines Substrates auf einen Compact-Flash-Speicherchip wurden realisiert. Eine erste Demultiplexerschaltung zum selektiven Lesen und Beschreiben von Probenträgern unter kryogenen Bedingungen wurde entwickelt [Ihmig 2006a]. Sie ergänzte die Entwicklung einer ersten prototypischen Lagerschublade[113], die zur Aufnahme von Substratstapeln mit integriertem Compact-Flash diente. Mit Hilfe der erwähnten Schaltung wurde der selektive Zugriff auf jeweils einen von acht Steckplätzen möglich, um Lese- und Schreibvorgänge auf die Speichermedien der Substratstapel auszuführen.

---

[110] Untersucht wurde NAND-Flash im Compact-Flash-Formfaktor mit integriertem Controller.

[111] Hier ist unter anderem die Einzelentnehmbarkeit von Aliquoten zu nennen.

[112] Kleinere Probenvolumina führen aufgrund geringerer Wärmekapazität zu reduzierter Eiskristallbildung.

[113] Die Probenschublade kann mit der in Kapitel 5.2 beschriebenen Applikation ‚Cryoplex' ausgelesen werden.

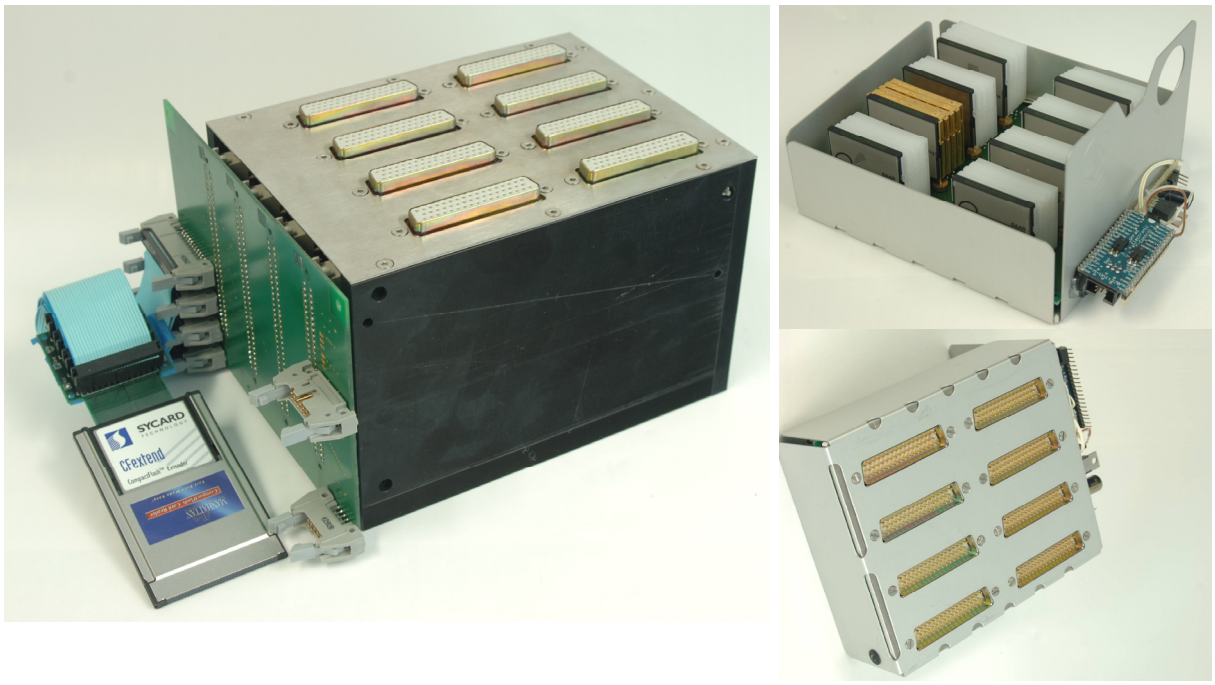

Abbildung 27: Links: eine prototypische Demultiplexer-Lesestation mit acht Steckverbindern zum Konnektieren der prototypischen Lagerschublade für Substratstapel mit integriertem Compact-Flash-Speicher. Zu sehen sind auch die Steckverbinder auf der Platine zur externen Ansteuerung der Schaltung. Die angeschlossene CardBus-Steckkarte dient zum Anschluß der Lesestation als Wechseldatenträger an einen Computer. Rechts oben: die prototypische Lagerschublade für Substratstapel. Sie verfügt über acht Steckplätze für die vertikale Aufnahme von Substratstapeln. Rechts unten: Bodenansicht der Lagerschublade. Zu sehen sind die acht Steckverbinder zum Konnektieren der Compact-Flash-Speicher mit den Steckverbindern der Lesestation.

Die oben beschriebenen Substratstapel wurden entwickelt, um eine sehr hohe Packungsdichte von Proben mit minimierten Volumina zu erreichen[114]. Innerhalb des GHRC-Konsortiums werden jedoch deutlich größere Probenvolumina benötigt, denn von HIV-relevanten Proben werden unter anderem Vollblut, Plasma und Serum in Röhrchen mit jeweils 2ml Volumen und Viruskulturen in Volumina zwischen 100µl und 500µl kryokonserviert. Da es auch möglich sein muß, einzelne Proben voneinander unabhängig zu lagern und zwischen verschiedenen Konsortien auszutauschen, wurden für GHRC diskrete Substrate mit größerem Volumen und jeweils eigenem Flash-Speichermedium realisiert[115][Ihmig 2008]. Als Speichermedien wurden serielle Flash-Speicherchips mit *Serial Peripheral Interface* (*SPI*) gewählt, da SPI der erforderlichen Flexibilität hinsichtlich der logischen Tiefe einer elektronischen Lagertank-Infrastruktur[116] genügt, zudem eine flexible modulare Ausstattung der einzelnen Hierarchieebenen erlaubt und nur wenige Leitungen benötigt, so daß der Wärmeeintrag in

[114] Bei einer Zellkonzentration von $1x10^7$ Zellen pro Milliliter sind in einem 25µl-Well etwa 25000 Zellen enthalten.

[115] Es wurden Prototypen diskreter Substrate verschiedener Volumina (2ml, 100 µl und 500µl) konstruiert. Die 2ml-Substrate wurden mittlerweile zur Serienreife weiterentwickelt.

[116] beschrieben im nächsten Abschnitt

Kryo-Lagertanks gegenüber parallelen Interfaces minimiert wird[117]. Folgende Abbildung (entnommen aus [PCT 2008]) zeigt das mittlerweile als Serienprodukt verfügbare GHRC-Substrat und illustriert seinen Aufbau:

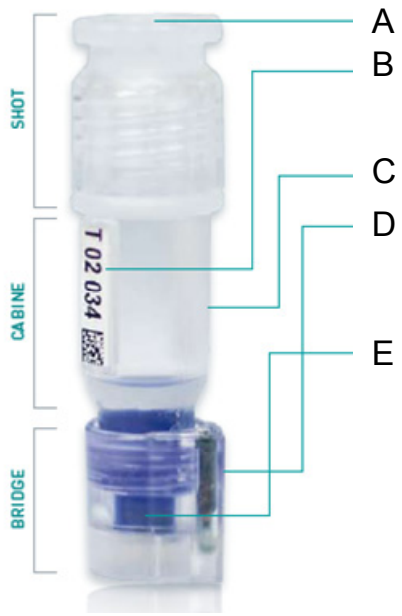

Abbildung 28: Als Serienprodukt erhältliches GHRC-Substrat (‚Icebreaker'), hergestellt von Perma Cryo Technologie GmbH. Das Substrat besteht aus drei Teilen: dem Schraubverschluß (‚Shot') mit Innengewinde, dem eigentlichen Probenbehälter (‚Cabine') und dem Sockel (‚Bridge') mit integriertem seriellem Flash-Speicherchip und Orientierungsnase. Folgende Identifizierungsmöglichkeiten sind integriert: (A) farbliche Codierbarkeit des Schraubverschlusses, (B) in Spritzgußtechnik eingearbeitetes Etikett mit eindeutiger Seriennummer und 2D-Barcode, (C) eingespritztes Etikett zur manuellen Beschriftung, (D) RFID-Tag zur berührungslosen Identifizierung und (E) serieller Flash-Speicherchip.

Um auch existierende Probensammlungen an die entstandene Technologieplattform anbinden zu können, wurden zusätzlich Adaptoren (‚Portlink') für Standard-Kryoröhrchen entwickelt. Sie umschließen ein Kryoröhrchen und werden mit einem Sockel mit integriertem seriellem Flash-Speicherchip versehen[118]. Der Adapter verfügt im Wesentlichen über die gleichen Identifizierungsmerkmale wie das GHRC-Substrat und wird mittlerweile ebenfalls in Serie hergestellt (Perma Cryo Technologie GmbH). Folgende Abbildung illustriert die Anwendung des Adapters:

[117] Ein weiterer Vorteil gegenüber Compact-Flash ist die direkte Ansteuerung des Flash-Speicherchips ohne zwischengeschalteten Controllerchip.

[118] Die Sockel von ‚Icebreaker' und ‚Portlink' sind gleich.

Abbildung 29: Anwendung des Adapters für Standard-Kryoröhrchen. Links: Standard-Kryoröhrchen mit einem Volumen von 2ml. Mitte: Adapter (‚Portlink', hergestellt von Perma Cryo Technologie GmbH) zum Integrieren von Standard-Kryoröhrchen in die Technologieplattform. Der Adapter verfügt im Wesentlichen über die gleichen Identifizierungsmerkmale wie das GHRC-Substrat und nutzt die gleichen Sockel. Rechts: adaptiertes Kryoröhrchen, umschlossen von einem ‚Portlink'.

Der Sockel stellt die Verbindung zwischen seriellem Flash-Speicherchip und Kryo-Lagerboxen dar, die dem Standard-Labor-Titerplattenformat entsprechen und zur Organisation[119], Lagerung und Transport von GHRC-Substraten und adaptierten Standard-Kryoröhrchen dienen:

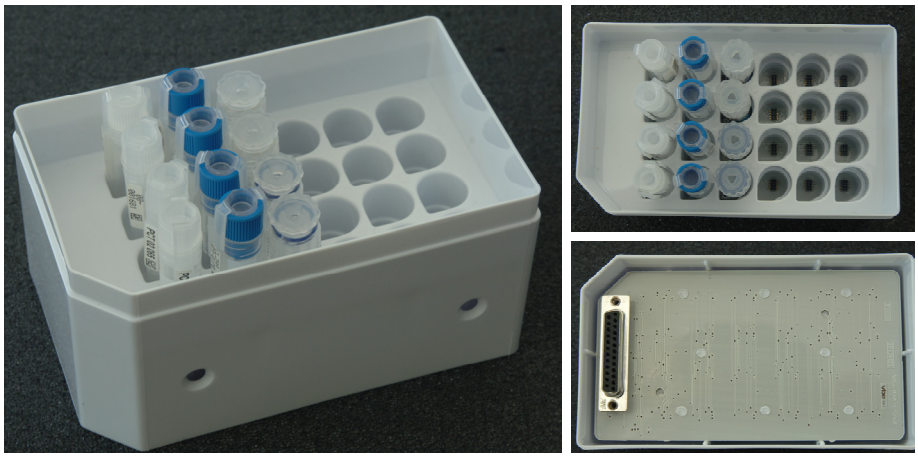

Abbildung 30: Verschiedene Blickwinkel auf eine Lagerbox für 24 GHRC-Substrate oder adaptierte Standard-Kryoröhrchen mit jeweils 2 ml Volumen. Die Positionen und Steckverbinder zur Aufnahme der Sockel sind in 6 Spalten und 4 Reihen angeordnet. Das elektronische PCB-Board dient als Interface zwischen seriellem Flash-Speicherchip der Substrate und elektronischer Tank-Infrastruktur und den in Kapitel 5 beschriebenen Dockingstationen.

[119] Die Organisation von Substraten und adaptierten Röhrchen auf Trägerplatten und in Lagerboxen ist auch hinsichtlich der in Kapitel 5 beschriebenen Ergebnisse bedeutsam.

Innerhalb der Technologieplattform stellen Lagerboxen die Hierarchieebene oberhalb von Proben bzw. Substraten dar. Jede Lagerbox enthält ein Printed Circuit Board (PCB), das als elektronische Schnittstelle zwischen den Speicherchips der Substrate und der elektronischen Lagertank-Infrastruktur fungiert. Das PCB-Board der Lagerbox beinhaltet eigene serielle Flash-Speicherchips, die unter anderem bei der Probenlagerung genutzt werden, aber auch im Zusammenhang mit den in Kapitel 5 beschriebenen Ergebnissen.

Die in diesem Abschnitt beschriebenen GHRC-Substrate, Adapter und Lagerboxen mit integriertem Flash-Speicherchip sind Teile der entwickelten Technologieplattform. Sie werden im Zusammenhang mit benötigten Datenstrukturen auch allgemein als *Cryo-Devices* bezeichnet. Gleiches gilt auch für die in Abschnitt 4.4 beschriebenen Komponenten der elektronischen Lagertank-Infrastruktur. Folgende Abbildung fasst der Vollständigkeit halber die wesentlichen Entwicklungsschritte auf dem Weg vom ersten prototypischen Schiebesubstrat bis zum Serienprodukt zusammen:

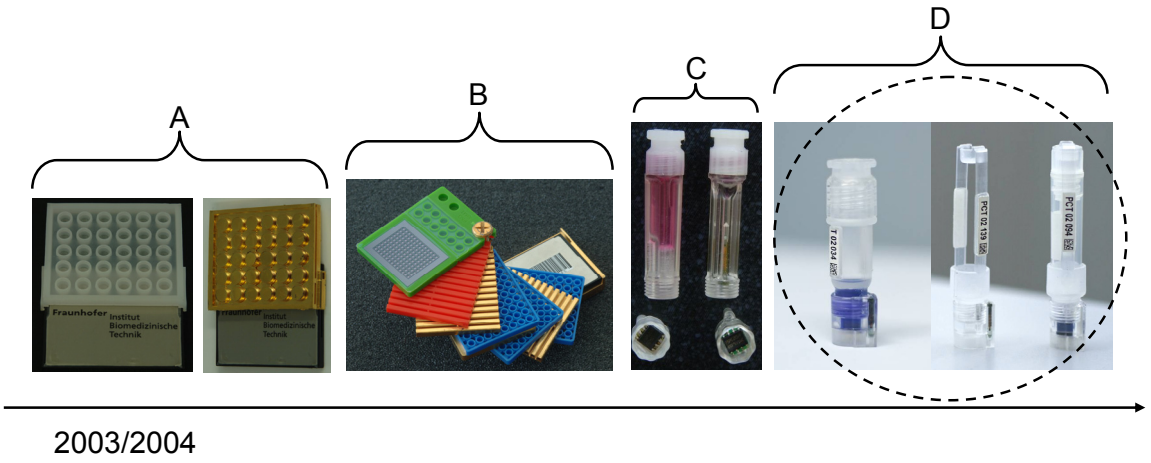

Abbildung 31: zusammengefaßte Historie der wesentlichen Substratentwicklungen. (A) zeigt die oben beschriebenen stapelbaren Substrate in Verbindung mit einem gemeinsamen Compact-Flash-Speicher. Sie waren Ausgangspunkt für die Entwicklung der Technologieplattform und für die weitere Substratentwicklung, beispielsweise für das in (B) illustrierte Multisubstrat. Verschiedenartige Substrate sind auf einer Achse aufgereiht und können nach Bedarf ausgeschwenkt werden. Die Achse dient auch zur Arretierung der Substrate. Hier aufgereiht sind ein Slilziumsubstrat mit 130 Probentöpchen zu jeweils 0,6µl in grüner Halterung, Fassungen für Schlauchsubstrate (rot, goldfarben) und Weiterentwicklungen des Schiebesubstrates mit ausstanzbaren einzelnen Probentöpfchen (blau). Ein Compact-Flash ist als gemeinsamer Speicher eingearbeitet. (C) zeigt prototypische diskrete Substrate mit einem Volumen von 500µl (links) und 100µl (rechts) und mit mittig eingespritztem RFID Tag (nur 100µl-Substrat). Jedes diskrete Substrat verfügt über einen eigenen seriellen Flash-Speicherchip, der in einem Schraubsockel untergebracht ist. Dieser Prototyp hat bereits die wesentlichen Merkmale des Serienproduktes. (D) zeigt die von Perma Cryo Technologie GmbH in Serie gefertigten diskreten GHRC-Substrate und Adapter für Standard-Kryoröhrchen.

4.4 Eine elektronische Lagertank-Infrastruktur

Um eine exakte Lokalisierung von Proben während ihrer Langzeitlagerung im Kryo-Lagertank zu erreichen und jederzeit auf die Speicherchips der Substrate zugreifen[120] zu können, wurde aufbauend auf Untersuchungen zur Tieftemperaturkompatibilität von Bauteilen [Zimmermann 2004] eine elektronische Infrastruktur für Kryo-Lagertanks entwickelt [Shirley 2009]. Diese ist hierarchisch aufgebaut und beinhaltet wohldefinierte Schnittstellen zwischen ihren Ebenen. Jede Komponente der elektronischen Infrastruktur wird als *Cryo-Device* bezeichnet und ist über eine serielle Schnittstelle mit der jeweils nächsthöheren logischen Ebene verbunden. Substrate und Adapter stellen die niedrigste Ebene der Hierarchie dar, Lagerboxen entsprechen der nächsthöheren Ebene. Jede Hierarchieebene ist mit einem oder mehreren seriellen Flash-Speicherchips ausgestattet. Oberhalb der Ebene der Substrate und Adapter dienen diese Speicherchips unter anderem dazu, Anzahl und Position untergeordneter Cryo-Devices zu verwalten und somit ein elektronisches Inventar bereitzustellen. Zusätzlich enthalten die Speicherchips die erforderlichen Informationen zur Ansteuerung der untergeordneten Cryo-Devices. Jede Komponente hält in ihren Speicherchips Informationen zu ihrer Identifizierung und ihren spezifischen Charakteristika[121] bereit. Die geschilderten Prinzipien wurden auf allen Ebenen der Hierarchie umgesetzt, so daß die elektronische Lagertank-Infrastruktur um neue Typen von Komponenten erweitert werden kann.

Der mechanische Rahmen für die elektronische Tank-Infrastruktur ist ein neuartiges Rack-System für Kryo-Lagertanks, das anstelle des konventionellen Racksystems[122] eingesetzt wird. Die Racks sind in Form von Quadranten realisiert, deren Gerüst aus Aluminium besteht. Sie enthalten jeweils eine *Backplane* [Fuchs 2008], die über Steckverbinder das modulare Konnektieren einzelner Lagerebenen (*Wings*) erlaubt. Die Backplane ist modular aufgebaut und verfügt über einen Mikrocontroller zum Steuern des Datenflusses über den SPI-Bus und zum autarken Ermitteln von Meßwerten. Als Mikrocontroller wird ein 32-bit RISC ‚embedded processor' verwendet. Die Backplane enthält zusätzlich einen Arbeitsspeicher zur Datenverarbeitung, bestehend aus 16MB statischem RAM, und zusätzlich einen 8MB großen parallelen Flash-Speicher zum Vorhalten von Lookup-Tabellen und Betriebssystem. Die Lookup-Tabellen werden für optimierten Betrieb der SPI-Flash-Speicher und für das Kalibrieren von Sensoren benötigt. Ein zusätzliches Analogmodul dient hauptsächlich zur Erzeugung der für analoge Messungen erforderlichen Betriebsströme und -Spannungen. Die elektronische Verbindung jedes Racks nach außen wurde mit Hilfe Teflon-isolierter Leitungen am oberen Ende der Racks realisiert. Die Backplane stellt innerhalb eines Lagertanks die höchste

---

[120] Dies ist relevant für gezielten Zugriff, aber auch für automatisiertes Aufzeichnen von Lagerbedingungen.

[121] Komponentenspezifische Datenstrukturen werden in Kapitel 5.6 konzipiert.

[122] beschrieben in Abschnitt 2.2.1

Hierarchieebene dar. Ihre Flash-Speicherchips dienen unter anderem als Inventarspeicher für die Inhalte der einzelnen Wings. Wings bestehen im Wesentlichen aus einer Leiterplatte mit Steckverbindern zur Aufnahme von Lagerboxen und werden über einen Ausschwing-Mechanismus zugänglich. Je nach verwendeten Röhrchen oder Probenboxen können die vertikalen Abstände zwischen einzelnen Wings so variiert werden, daß auch eine Kompatibilität mit künftigen Substraten (vom Mikrosubstrat bis hin zu Blutbeuteln) erreicht wird. Folgende Abbildung zeigt ein Rack mit seinen Komponenten:

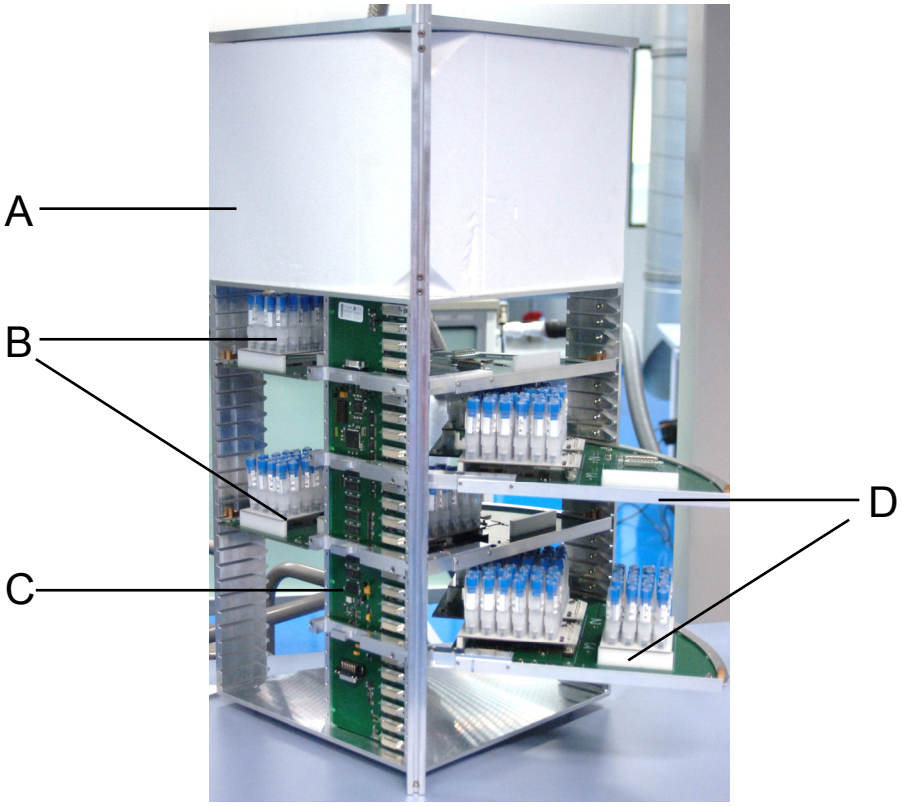

Abbildung 32: Ein Rack mit seinen Komponenten. Das abgebildete Rack ist mit vier modularen Lagerebenen (Wings) ausgestattet. Jedes Wing und die Backplane verfügen über eigene Temperatursensoren. (A) Thermische Isolierung an der Rack-Oberseite, (B) adaptierte Probenröhrchen, aufgesteckt auf Lagerbox-PCB-Boards, die mit Lagerebenen (Wings) konnektiert sind. Diese Wings sind eingeklappt. (C) Backplane mit Mikrokontroller und Inventarspeicher, (D) ausgeklappte Wings mit Lagerbox-PCB-Boards, auf denen adaptierte Probenröhrchen organisiert sind.

Der Mikrocontroller der Backplane kann unter Verwendung von Demultiplexern auf den verschiedenen Hierarchieebenen bis hinab auf den Speicherchip jedes einzelnen Röhrchens zugreifen. Dazu nutzt er die Informationen über die einzelnen Hierarchiekomponenten, die in ihren Speicherchips abgelegt sind. Nach außen hin erscheint jedes Rack als ein einziges serielles Gerät. Der Mikrocontroller kann auch Temperatursensoren von Cryo-Devices abfragen, um die Lagerbedingungen zu dokumentieren. Außerdem steuert er das Power Management der Wings, Boxen und Röhrchen. Automatische Funktionen des Racks sind unter anderem:

- Periodisches Abfragen von Temperatursensoren
- Aktualisieren der Temperatur-Historie der einzelnen Probenröhrchen
- Erstellen eines Inventars
- Überprüfen korrekter Platzierung oder korrekter Entnahme einer Probe

Aufgrund der Ausführung als Quadranten können maximal vier Racks in einem Kryotank verwendet werden. Eine spezielle Haubentechnik schirmt die Racks gegenüber der Umgebungsluft ab und etabliert eine trockene, kalte Atmosphäre aus Stickstoffgas. Dies verhindert die Kondensation von Luftfeuchtigkeit und somit Eisbildung und Korrosion. Außerdem wird die Kühlkette erhalten und Kondensation von Sauerstoff aus der Luft verhindert, die ansonsten zum Ansammeln flüssigen Sauerstoffs im Lagertank führen würde. Die Haube enthält Liftsysteme für das Anheben und Absenken der Racks. Eine Schleuse erlaubt das Einbringen und Entnehmen von Proben. Kryolagertank und Haube bilden einen Probenlagerturm, dessen Aufbau in folgender Abbildung illustriert wird:

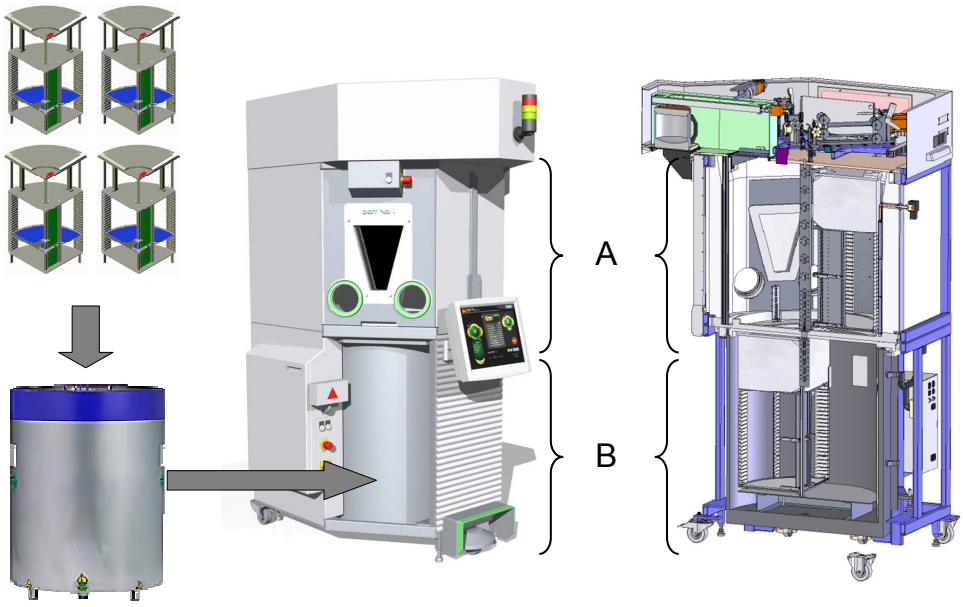

Abbildung 33: Illustration eines IBMT-Probenlagerturms anhand von CAD-Modellen (Außenansicht und Querschnitt). (A) entspricht der beschriebenen Haube, die auf einem Kryo-Lagertank (B) aufsetzt und das Innere des Lagertanks von der Umgebungsluft abschirmt. Dadurch werden Kondensation von Luftfeuchtigkeit und damit Eisbildung auf den Komponenten der Infrastruktur vermieden. Zusätzlich wird die Kühlkette gesichert. In der Haube integriert sind Liftsysteme für das Anheben und Absenken der einzelnen Racks bei Bedarf. Eine Schleuse erlaubt das Einbringen und Entnehmen von Proben bei Aufrechterhaltung der Schutzatmosphäre. Außerdem in der Haube enthalten sind Sichtfenster und Eingriffe für die Hände, um Proben auf die Lagerebenen der Racks platzieren oder sie von dort entnehmen zu können. Ein Computer erlaubt die Bedienung der Liftsysteme, der Schleuse und die Kontrolle der elektronischen Infrastruktur, beispielsweise um eine Abfrage von Lagerplätzen durchzuführen oder ein vollständiges Inventar zu generieren. Es exisitieren verschiedene prototypische Generationen des Probenlagerturms. Die Serienfertigung wird durch einen Industriepartner durchgeführt (Askion GmbH).

4.5 Formulierung des eigenen Ansatzes

Der wissenschaftliche Wert von Probensammlungen ist einerseits von der Qualität der Probenlagerung und von der Qualität des Umgangs mit vorhandenem Probenwissen[123] abhängig, andererseits jedoch durch die Qualität der Proben selbst und durch die Qualität des zugehörigen Probenwissens limitiert. Der Ansatz des Fraunhofer-IBMT, eine angemessen hohe Qualität im Biobanking durch dezentrale Speicherchips bei der Probe zu erreichen, fokussierte ursprünglich nur auf optimierte Bewahrung vorhandenen deklarativen Probenwissens und auf einen vereinfachten Wissensaustausch. Um zusätzlich eine hohe Qualität von Probe und Probenwissen sicherzustellen, sind qualitätssichernde Maßnahmen bereits bei der Probenbearbeitung und bei der Generierung und Akquise von Probenwissen erforderlich.

Probenwissen entsteht[124] beispielsweise bei der Entnahme einer Probe aus einem Organismus, bei ihrer Verarbeitung oder ihrer Untersuchung hinsichtlich relevanter Fragestellungen[125]. Die Qualität von Probe und Probenwissen resultiert unmittelbar aus der Bearbeitungsqualität und der Handlungsgüte bei der Ausführung biomedizinischer Protokolle[126]. Speziell die Qualität von Probenwissen ist zusätzlich von der Qualität der Wissensakquise[127] abhängig. In Kapitel 3.1 wurden die in biomedizinischen Forschungslaboratorien üblichen Arbeitsweisen hinsichtlich der Ausführung und Dokumentation von Laborprotokollen beschrieben und einige dabei vorhandene Probleme aufgezeigt, die sich negativ auf die Qualität von Proben und zugehörigem Wissen auswirken können. Nachfolgend werden die wesentlichen Probleme nochmals zusammenfassend aufgelistet:

- Biomedizinische Forschungslaboratorien sind mit einer Vielzahl unterschiedlichster Laborprotokolle für verschiedenste Zelltypen konfrontiert. Daraus entsteht eine reale Verwechselungsgefahr bei der Probenbearbeitung[128], bei der Generierung und der Akquise von Probenwissen. Diese Verwechslungsgefahr wird durch die papiergebundene

---

[123] Dies entspricht dem Aspekt der Wissensrepräsentation gemäß dem Prozeßmodell nach [Reinmann-Rothmeier 2000].

[124] Die Entstehung von Probenwissen entspricht dem Aspekt der Wissensgenerierung gemäß dem Prozeßmodell von [Reinmann-Rothmeier 2000] (siehe Definition in Kapitel 2.1).

[125] Probenwissen entsteht auch während Lager- oder Transportvorgängen und kann beispielsweise in Form von Temperaturhistorien erfaßt werden, um eine Korrelierung mit biologischen Eigenschaften zu ermöglichen.

[126] Somit ist die Qualität von Probe und Probenwissen immer dann stark von individuellem implizitem Wissen abhängig, wenn erfahrungsgeleitetes Arbeitshandeln stattfindet.

[127] Wissensakquise meint hier das Aufzeichnen von Labordokumentation, also prozeduralen Wissens über die einzelnen Bearbeitungsschritte und generierten deklarativen Probenwissens.

[128] Probenbearbeitung anhand der Ausführung von Protokollen entspricht dem Aspekt der Wissensnutzung im Sinne des Prozeßmodells.

Arbeitsweise und durch die beschriebene referenzierende Zuordnungsmethodik verstärkt.

- Es ko-existieren häufig verschiedene Versionen desselben Laborprotokolls, die mindestens in Teilen voneinander abweichen. Dies kann funktionale Auswirkungen auf Proben und Auswirkungen auf entstehendes deklaratives Probenwissen haben.
- Laborprotokolle haben keine definierte Detailschärfe. Das Fehlen oder die Unvollständigkeit expliziten prozeduralen Wissens führt zum Übergang in einen erfahrungsgeleiteten Handlungsmodus. Dieser ist durch individuelles implizites Wissen geprägt, das eventuell fehlerhaft ist.
- Unterschiedliches Arbeitshandeln, uneinheitliche Handlungsgüte und abweichende Bearbeitungsgeschwindigkeiten oder -dauern beeinflussen die Qualität von Probe und entstehendem Wissen.
- Unterschiedliche Detailschärfe, Sorgfalt und Zeitnähe zur Probenbearbeitung führen bei der Wissensakquise zu qualitativen Unterschieden bezüglich entstandenem Probenwissen und dokumentiertem Prozeßwissen.
- Die Übertragung handschriftlich akquirierter Dokumentation (deklaratives Probenwissen und prozedurales Prozeßwissen) in ein elektronisches System stellt eine potentielle Fehlerquelle dar.

In Reihenfolge der aufgelisteten Probleme ergeben sich die nachfolgenden Anforderungen. Sie müssen erfüllt werden, um die Qualität von Proben und Probenwissen in biomedizinischen Forschungslaboratorien zu gewährleisten:

- Ausschließen von Verwechselungen (hinsichtlich Protokollzuordnung, Protokollausführung, Wissensgenerierung, Wissensakquise und Bewahrung von Probenwissen und Labordokumentation)
- einheitliche und somit vergleichbare Abarbeitung von versionierten Protokollen
- Minimierung des Einflusses von individuellem implizitem Wissen
- einheitliche und unmittelbare Akquise von generiertem Wissen
- Dokumentieren von Bearbeitungsdauern für eine spätere Korrelation mit eventuell auftretenden biologischen Effekten

Das Streben nach hoher Bearbeitungsqualität, hoher Präzision, Effizienz und Reduzierung menschlicher Fehler hat im Kontext sehr spezialisierter Laboratorien mit entsprechendem Probendurchsatz zur Entwicklung der in Kapitel 3.3 beschriebenen Laborautomatisierungsansätze und der dazu komplementären Laborinformations-Managementsysteme geführt. Das Risiko von Verwechselungen durch referenzierende Zuordnungsmethodik prozeduralen und deklarativen Wissens ist allerdings auch bei solchen Systemen existent. Die Forschung an unterschiedlichen Probentypen, die große Anzahl verschiedener Protokolle, der geringe Automatisierungsgrad und der gleichzeitig hohe Anteil an manuellen Bearbeitungsschritten erlaubt

keinen effizienten Einsatz der klassischen Automatisierungsansätze in biomedizinischen Forschungslaboratorien[129]. Vielmehr ist ein anderer Ansatz erforderlich, der dem Charakter biomedizinischer Forschungslaboratorien gerecht wird und die erforderliche Flexibilität aufweist.

Der im folgenden formulierte eigene Ansatz (*ChameleonLab-Prinzipien*[130]) vertieft eine Idee aus [Durst 2003], um den geschilderten Problemen hinsichtlich der Qualität von Probenbearbeitung, von dabei generiertem Wissen und von Wissensakquise in biomedizinischen Forschungslaboratorien zu begegnen und die oben identifizierten Anforderungen zu erfüllen.

1) *Kopplung von Probe und Protokoll*: zusätzlich zu deklarativem Probenwissen soll der Speicherchip des Substrates das für die Präparation einer Probe erforderliche prozedurale Wissen enthalten. Diese Vorgehensweise ersetzt die referenzierende Zuordnungsmethodik und schließt fehlerhafte Zuordnungen von Protokollen aus. Prozedurales Wissen kann dann gemeinsam mit der Probe langzeitgelagert werden (Wissensbewahrung nach [Probst 1998], siehe Kapitel 2.1) oder im Fall von Kooperationen mit anderen Laboratorien ausgetauscht werden (Wissenskommunikation nach [Reinmann-Rothmeier 2000] und Wissensverteilung nach [Probst 1998]).

2) *Probe steuert ihre eigene Präparation*: das auf dem Speicherchip des Substrates gespeicherte prozedurale Wissen soll die Bearbeitung der Probe steuern. Dies kann mit Hilfe eines Informationssystems erfolgen, das ein einheitliches und detailliertes explizites Handlungswissen für manuelle, semimanuelle und automatisierte Bearbeitungsschritte zur Verfügung stellt und dadurch aktiv einem erfahrungsgeleiteten Handlungsmodus mit seinen Risiken (siehe Kapitel 2.1) entgegenwirkt. Gemäß den Ausführungen in Abschnitt 2.2.4 sind Laborprotokolle aus wirtschaftsinformatischer Sicht Workflows, deren Ausführung durch geeignete Workflow-Management-Systeme[131] unterstützt werden kann. Das prozedurale Wissen kann also in Form einer elektronischen Workflow-Definition auf dem Speicherchip des Substrates repräsentiert sein, gemäß Abschnitt 2.2.4 bestehend aus einzelnen Aktivitäten, die jeweils einen Bearbeitungsschritt des Protokolls repräsentieren.

3) *Kopplung von Probe und Labordokumentation*: die Akquise von während der Bearbeitung generiertem deklarativem und prozeduralem Wissen soll direkt in elektronischer Form auf dem Speicherchip des Substrates erfolgen. In der Workflow-Definition können zu diesem Zweck die vom Anwender einzugebenden oder von Geräten abzufra-

---

[129] Gemäß der Begriffsdefinition in Kapitel 3.1.

[130] Der Begriff *ChameleonLab* ist ein proprietärer Begriff, der im Rahmen dieser Arbeit entstanden ist. Er soll die Flexibilität und Anpassung des Systems an die Probe zum Ausdruck bringen.

[131] Dies deckt sich mit den in Abschnitt 2.2.4 aufgelisteten Zielen (Vereinheitlichung von Arbeitsabläufen, Verbesserung der Qualität von Arbeitsabläufen, Verkürzung von Bearbeitungszeiten, Erhöhung von Informationsverfügbarkeit) beim Einsatz von WMS und mit ihrer Funktionalität, benötigte Daten zur Verfügung zu stellen.

genden Daten definiert werden, die dann bei der Abarbeitung durch das Workflow-Management-System akquiriert werden sollen. Es sollen automatisch Zeitmarken und Bearbeitungsdauern für jeden Bearbeitungsschritt akquiriert werden. Das WMS soll auf diese Weise ein einheitlich strukturiertes und vergleichbares Wissen aufzeichnen und Funktionalitäten eines elektronischen Laborbuches erfüllen. Die Wissensakquise direkt auf den Speicherchip des Substrates ersetzt papiergebundene Dokumentation und deren Zuordnung über referenzierende Verfahren. Bewahrung und Kommunikation des generierten Wissens wird vereinfacht und Verwechslungen werden ausgeschlossen.

Die Umsetzung der formulierten Prinzipien soll die Qualität von Proben und von Probenwissen bereits bei Probenbearbeitung und Wissensgenerierung sicherstellen. Dadurch wird der Ansatz des Fraunhofer-IBMT, eine angemessen hohe Qualität im Biobanking durch Kopplung von Probe und elektronischem Speicherchip zu erreichen, vervollständigt.

5 Ergebnisse

Im Wesentlichen beschreibt dieses Kapitel die unter wissenschaftlichen Gesichtspunkten erfolgte Umsetzung des in Kapitel 4.5 beschriebenen eigenen Ansatzes. Diese ist parallel zu und in enger Interaktion mit der Entwicklung der in Kapitel 4.3 beschriebenen Technologieplattform erfolgt, aber auch in enger Abstimmung mit repräsentativen Biologen und Laborpersonal unter Berücksichtigung der von ihnen kommunizierten Anforderungen. Zudem wurden Ergebnisse der Diplomarbeit (vgl. [Durst 2003]) als Ausgangsbasis für die Definition von Datenstrukturen verwendet, die zur dezentralen Wissensrepräsentation auf elektronischen Speicherchips benötigt werden. Ursprünglich als dezentrale Backups von deklarativem probenbezogenem Wissen konzipiert, wurden diese Datenstrukturen im Rahmen der vorliegenden Arbeit anhand der fortschreitenden Entwicklung von Probenträgern und Lagertechnologie verändert und um prozedurales Wissen für die Probenpräparation, um deklaratives Wissen für den Betrieb der Lagertechnologie und um geeignete Strukturen für die Akquise von solchem Wissen erweitert, das bei Präparation und Lagerung generiert wird.

Im Rahmen der vorliegenden Arbeit sind zwei Versionen von ‚ChameleonLab' entstanden:

- ein erstes Testsystem
- der GHRC-Prototyp

Die folgenden Kapitel beschreiben die Entwicklung beider Versionen vor dem Hintergrund repräsentativer Laborprotokolle, der Entwicklung der Technologieplattform, der Anforderungen an dezentrale Backups und der Interaktion mit Probendatenbanken.

5.1 Ein erstes Testsystem

Ein wichtiger Schritt auf dem Weg zur vollständigen Umsetzung der in Kapitel 4.5 formulierten ChameleonLab-Prinzipien war die Entwicklung eines ersten Testsystems, das hauptsächlich auf Benutzerführung durch ein Worfklow-Management-System[132] bei der Abarbeitung biomedizinischer Laborprotokolle fokussierte. Dieses System sollte dem Laborpersonal einheitliches prozedurales Wissen elektronisch dort zur Verfügung stellen, wo es benötigt wird[133] und somit die übliche Verwendung von Laborprotokollen in Papierform ersetzen. Das System sollte als konkrete Grundlage für die Evaluierung grundsätzlicher Machbarkeit, Akzeptanz durch potentielle Benutzer und zur Identifizierung weiterer Anforderungen an Fol-

[132] Ziele sind einheitliche Bearbeitungsqualität, Handlungsgüte und Akquise vergleichbaren Wissens.

[133] Hiermit ist gemeint, daß die an einem bestimmten Arbeitsplatz im Labor erforderlichen Informationen auch genau dort zur Verfügung stehen.

gesysteme dienen. Im Zusammenhang mit dieser Intention konnte das Prinzip der verwechselungssicheren Protokollzuordnung (*Kopplung von Probe und Protokoll*) zunächst vernachlässigt werden. Dieser Sachverhalt war aus zwei Gründen vorteilhaft: zum einen konnte ein existierendes System aus dem biomedizinischen Bereich als Basis für die Implementierung des Testsystems verwendet werden, unter Beibehaltung seiner Mechanismen zur Workflowdefinition und seiner Datenstrukturen für die Workflow-Repräsentation. Zum anderen konnte die Implementierung des Testsystems unabhängig von der Erforschung konkreter Speicherchips und ihrer Eigenschaften[134] erfolgen.

5.1.1 Abstraktion repräsentativer biomedizinischer Laborprotokolle

Um die wesentlichen Anforderungen an ein Testsystem zu identifizieren, werden verschiedene Laborprotokolle betrachtet, analysiert und ihre Eigenschaften abstrahiert. Es folgen beispielhaft zwei repräsentative Laborprotokolle für vorbereitende Präparation[135] in ihrer Original-Formulierung[136].

Protokoll zum Einfrieren einer L929-Zellsuspension (Mausfibroblasten)

- Nach der Lieferung der Zellen: Zwischenlagerung im Brutschrank (37°C)
- Behälter mit Suspension in Laminar Flow (Werkbank) stellen
- Zellzahl und Vitalität bestimmen
- Zellen abzentrifugieren mit 1000 U/min 5 Minuten, dann Überstand abnehmen, frisches Medium hinzufügen und resuspendieren. Optimale Konzentration liegt im Bereich 1x10*6 bis 5x10*6 Zellen /ml.
- Kryoprotektanzien (DMSO) und deren Konzentration wählen
- Die Zellen in einem Röhrchen in Plotter bringen, DMSO in anderem
- Plotter programmieren, dass DMSO tropfenweise dazugegeben wird
- Kryosubstrate mit Plotter befüllen mit Zellsuspension und DMSO
- Die Substrate mit den Zellen mit Folien verschließen
- Programmiertes Kryogerät (Planer) einstellen: Einfrierprogramm wählen, die Daten über die konservierten Zellen speichern, das Programm starten
- Den Platz im Stickstoffbehälter suchen und reservieren
- Nach dem Einfrieren mit Transportbehälter die Kryosubstrate in Stickstofftank übertragen

---

[134] Zum Zeitpunkt der Entwicklung des Testsystems waren die Untersuchungen zur Kryokompatibilität von Speicherchips noch nicht abgeschlossen und daher kein entsprechender technologischer Rahmen für eine Kopplung von Probe und Protokoll gegeben.

[135] Siehe dazu Abschnitt 2.2.1.

[136] Die nicht definierte Detailschärfe von Protokollen (siehe Kapitel 2.2) wird bereits anhand dieser beiden Protokolle deutlich. Protokoll 1 weist einen sehr geringen Detaillierungsgrad auf und erfordert daher umfangreiche Substitution durch erfahrungsgeleitetes Arbeitshandeln.

Protokoll zum Einfrieren von hämatopoetischen Stammzellen mit FICOLL-Gradient

- EDTA Blutröhrchen 8min bei 400 G ohne Bremse zentrifugieren
- Überstand fast abnehmen (Serum verwerfen oder später weiterverwenden)
- verbleibendes Blut auf 11ml auffüllen mit Serum (im gleichen Röhrchen)
- 4ml Ficoll in kleines Falcon Röhrchen vorlegen
- 11 ml Blut vorsichtig (Röhrchen schräg halten) auf Ficoll zupipettieren
- 25-30min bei 400g ohne Bremse oder kleinster Bremse zentrifugieren
- Serumüberstand mit 1000er Pipette abnehmen und verwerfen
- Ring (enthält die Stammzellen) mit der Pipette abnehmen
- in neues Röhrchen überführen
- Leukozyten-Ring in kaltem PBS resuspendieren (4°C) 5 ml PBS
- 8 min bei 400g bei 4°C zentrifugieren (Bremse aus)
- Überstand verwerfen
- Pellet in kaltem PBS resuspendieren
- 8 min bei 400g bei 4°C zentrifugieren (Bremse aus)
- Überstand verwerfen
- Zellen in RPMI-Medium mit mind 10% besser 20% autologem oder humanen AB-Serum (hitzeinakiviert) und 10% DMSO resuspendieren und einfrieren. Die Prozentangaben beziehen sich auf das Endvolumen.

Es lassen sich folgende gemeinsame Eigenschaften abstrahieren, denen das Testsystem genügen muß:

- viele rein manuelle Präparationsschritte
- semimanuelle Präparationsschritte (Benutzer bedient Gerät)
- manuelle Transportvorgänge / manuelles Probenhandling
- rein sequentielle Ausführung von Präparationsschritten
- keine Kontrollstrukturen oder Bedingungen vorhanden

Bereits anhand dieser beiden Protokolle wird deutlich, daß verschiedene Protokolle hinsichtlich der involvierten Laborgeräte variieren und die verwendeten Laborressourcen (Arbeitsplätze, Laborgeräte) in unterschiedlicher Reihenfolge benutzt werden. Eine einheitliche Prozeßstrecke für biomedizinische Forschungslaboratorien ist daher nicht möglich[137], stattdessen müssen die Laborressourcen flexibel genutzt werden können. Weitere Anforderungen an das Testsystem wurden anhand des voraussichtlichen operativen Geschäftes künftiger Biobanken gemeinsam mit potentiellen Systembenutzern definiert. Dazu zählt unter anderem die Erwartung unterschiedlichster Probentypen, aus der sich die Anforderung der Systemkompatibilität mit vielen verschiedenen Präparationsprotokollen ergibt.

[137] Dies gilt auch besonders im Kontext der kontinuierlichen Optimierung von Protokollen.

5.1.2 Konzeption des Testsystems

In den Kapiteln 3.3 und 3.4 werden verschiedene Ansätze für Laborautomatisierung beschrieben, deren Entwicklung von der Anforderung nach hoher Effizienz und verbesserter Qualität bei der Bearbeitung eines hohen Probenaufkommens motiviert ist. Laborautomatisierungssysteme werden unter anderem im pharmazeutischen High-Throughput-Screening auf der Suche nach relevanten Wirkstoffen eingesetzt, häufig in Form modularer Workcells (siehe Abschnitt 3.3.2.2)[138]. Die Interaktion ihrer Komponenten (Laborgeräte, konsolidierte Laborgeräte und Robotik) wird durch einen System Controller (siehe Abschnitt 3.4.1) gesteuert. Als System Controller werden meist Scheduler (siehe Abschnitt 3.4.1.2) verwendet [Berman 2007]. Die Geräteintegration kann monolithisch im System Controller erfolgen (siehe Abschnitt 3.4.2.1) oder mit Hilfe von Device Integration Frameworks (siehe Abschnitt 3.4.2.2). Bei Verwendung von Device Integration Frameworks erfolgt die Geräteansteuerung durch periphere Softwarekomponenten, beispielsweise GUI-Geräteapplikationen oder Gerätetreiber, die mittels definierter Schnittstellen mit dem System Controller kommunizieren und gerätespezifische Kommandos mit gerätespezifischen Parametern[139] an das jeweilige Laborgerät senden. Ein als System Controller verwendeter Scheduler beispielsweise synchronisiert die anstehenden Bearbeitungsschritte (*Tasks* in der Nomenklatur der Laborautomatisierung, *Aktivitäten* im wirtschaftsinformatischen Zusammenhang) so, daß die Gerätekommandos exakt zum richtigen Zeitpunkt und in der richtigen Reihenfolge durch die entsprechenden Geräte ausgeführt werden. Diese Analogie zu der in Kapitel 5.1 als wesentliche Funktionalität des Testsystems definierten gezielten elektronischen Verteilung prozeduralen Wissens an Laborpersonal motivierte die Verwendung eines existierenden Schedulers aus dem pharmazeutischen High-Throughput-Screening als Ausgangsbasis für die Entwicklung des Testsystems.

Im Folgenden wird die Eignung des dynamischen[140] Schedulers ‚Bernstein' vor dem Hintergrund der festgestellten Anforderungen an das Testsystem diskutiert. ‚Bernstein' ist Teil der Laborautomatisierungssoftware ‚Evoscreen' (Evotec Technologies GmbH, Hamburg) und wird zur Steuerung verschiedener HTS-Module (unter anderem der in Abbildung 20 gezeigten Module ‚Scarina' und ‚Mitona' [Gentsch 2000]) genutzt.

---

[138] Modulare Workcells im *High-Throughput-Screening* (*HTS*) werden auch als *HTS-Module* bezeichnet.

[139] Dies entspricht gerätekompatiblem prozeduralem Wissen.

[140] Siehe Abschnitt 3.4.1.2.

‚Evoscreen' ist modular[141] aufgebaut und besteht im Wesentlichen aus dem Scheduler ‚Bernstein' als System Controller, aus peripheren Softwarekomponenten auf Gerätelevel zur Steuerung von Geräten und Robotik, aus einer zentralen Datenbank für Workflow-Definitionen (*Runs* in der ‚Evoscreen'-Terminologie) und aus diversen Editoren. Die Kommunikation zwischen ‚Bernstein' und den Geräteapplikationen erfolgt mittels einer definierten Schnittstelle über DCOM ([Chen 1998]). Die auf Gerätelevel abzuarbeitenden Prozesse werden unter Berücksichtigung von Events durch ‚Bernstein' synchronisiert. ‚Evoscreen'-Workflow-Definitionen bestehen aus rein sequentiell angeordneten Aktivitäten. Dieser Aufbau entspricht der Struktur der in Abschnitt 5.1.1 untersuchten Protokolle. Eine weitere Übereinstimmung ist das Fehlen von Kontrollstrukturen. Die ‚Evoscreen'-Workflow-Beschreibungssprache ist proprietär und auf die Bedürfnisse im High-Throughput-Screening ausgerichtet. Sie enthält beispielsweise Kommandos und Parameter, um Tasks für das Dispensieren von Suspensionen oder für das Inkubieren von Titerplatten zu definieren. Eine ‚Evoscreen'-Workflow-Definition ist wie folgt hierarchisch aufgebaut:

- Eine ‚Evoscreen'-*Aktivität* (oder *Task* bzw. *Step*) repräsentiert das für einen einzelnen Bearbeitungsschritt benötigte prozedurale Wissen in geräteverständlicher Form, d.h. sie besteht aus einem Gerätekommando und den benötigten Parametern.

- In einem ‚Evoscreen'-*Prozeß*[142] werden diejenigen Aktivitäten in totaler Ordnung zusammengefasst, die in unmittelbarer Abfolge durch dasselbe Laborgerät abzuarbeiten sind. Jeder ‚Evoscreen'-Prozeß besteht aus mindestens einer Aktivität.

- In einer ‚Evoscreen'-*Prozeßsequenz* werden Prozesse in totaler Ordnung gemäß der Reihenfolge ihrer Abarbeitung angeordnet.

- Eine ‚Evoscreen'-*Workflow-Definition (Rundefinition)* ist eine statische Zuordnung der einzelnen Prozesse einer ‚Evoscreen'-Prozeßsequenz zu den konkreten Geräten eines Modultyps, die bereits zum Entwurfszeitpunkt des Workflows erfolgt[143].

---

[141] Die modulare Architektur von ‚Evoscreen', die eine Interaktion zwischen System Controller und peripheren Geräteapplikationen mittels definierter Schnittstellen realisiert, ist vorteilhaft für die Implementierung des Testsystems. ‚Bernstein' kann ohne Änderungen verwendet werden, wenn die Komponenten des Testsystems seine Schnittstellen nutzen.

[142] Dies entspricht beispielsweise einem *Activity Set* in XPDL.

[143] Die Zuordnung der einzelnen Evoscreen-Prozesse zu den Geräten eines HTS-Modultyps erfolgt also bereits in der Evoscreen-Workflow-Definition. Insofern unterscheidet sich die Arbeitsweise von ‚Bernstein' von der üblichen Arbeitsweise statischer und event-basierter Scheduler, die selbst den Arbeitsplan berechnen. Diese Eigenschaft ist jedoch im Zusammenhang mit der Implementierung des Testsystems von besonderer Bedeutung.

Die Struktur von ‚Evoscreen'-Workflow-Definitionen (Runs) und die Synchronisierung der einzelnen Prozesse durch den Scheduler ‚Bernstein' ist im Folgenden abgebildet:

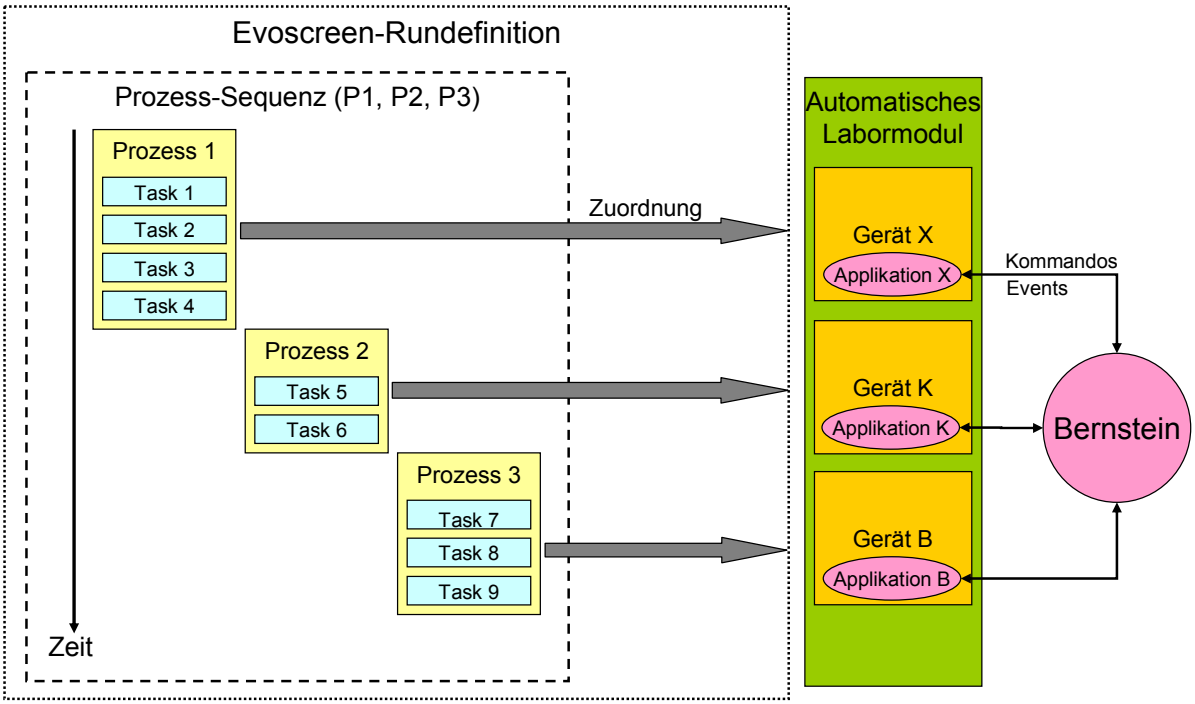

Abbildung 34: Schematische Darstellung der Struktur von ‚Evoscreen'-Workflow-Definitionen und der Synchronisierung der Prozesse durch den Scheduler ‚Bernstein'. Das für einen einzelnen Bearbeitungsschritt benötigte prozedurale Wissen wird in geräteverständlicher Form als Task (bzw. Step) repräsentiert. Ein Task besteht aus einem Gerätekommando und benötigten Parametern. Alle Tasks, die in direkter Folge durch dasselbe Gerät ausgeführt werden sollen, werden in Form eines ‚Evoscreen'-Prozesses total angeordnet. Eine ‚Evoscreen'-Prozeßsequenz definiert eine totale Ordnung auf Prozessen gemäß der Reihenfolge ihrer Abarbeitung. Eine ‚Evoscreen'-Workflow-Definition (oder Rundefinition) ist eine statische Zuordnung der einzelnen Prozesse einer ‚Evoscreen'-Prozeßsequenz zu den konkreten Geräten eines Modultyps. Diese Struktur wird mit Hilfe entsprechender Tabellen in der ‚Evoscreen'-Workflowdatenbank realisiert. In der Abbildung sind drei Prozesse mit ihren jeweiligen Tasks unterschiedlichen Geräten eines Labormoduls zugeordnet. Vorteil der statischen Zuordnung von Prozessen zu Geräten ist die Möglichkeit gezielter Wissensverteilung. Die Abarbeitung der einzelnen Prozesse wird durch Kommandos von ‚Bernstein' an gerätespezifische Applikationen synchronisiert. ‚Bernstein' kommuniziert bidirektional mit Hilfe definierter Schnittstellen und reagiert auf Events von Seiten der Geräteapplikationen.

Die Abarbeitung einer Instanz einer ‚Evoscreen'-Workflow-Definition erfolgt durch Interaktion des Schedulers ‚Bernstein' mit den involvierten Geräteapplikationen. Die Workflow-Definition referenziert auf die Definitionen der einzelnen ‚Evoscreen'-Prozesse. ‚Bernstein' verteilt diese Referenzen an die Geräteapplikationen, die daraufhin ihnen zugeordnete ‚Evoscreen'-Prozeß-Definitionen aus der ‚Evoscreen'-Datenbank laden. Die Synchronisie-

rung der Prozesse[144] einer ‚Evoscreen'-Workflow-Instanz ist event-basiert. Seien beispielsweise in obiger Abbildung die Prozesse 1 und 3 Prozesse zur Bearbeitung von Proben auf Mikrotiterplatten, sei Prozeß 2 ein Transportprozeß durch einen Roboter. Wenn beispielsweise die Ausführung von Prozeß 1 durch Gerät X abgeschlossen ist, erfolgt eine Meldung der Geräteapplikation X an ein Kommunikationsthread (siehe Abschnitt 3.4.1.2) von ‚Bernstein'. ‚Bernstein' sendet dann das Startsignal für die Ausführung von Prozeß 2 an Geräteapplikation K, deren Tasks einen robotisierten Transport der Mikrotiterplatte zu Gerät B durchführen. Ist der Transportprozess abgeschlossen, erfolgt eine Meldung der Geräteapplikation K an ‚Bernstein', der dann die Ausführung von Prozeß 3 an Gerät B startet, während an Gerät X die Bearbeitung einer weiteren Mikrotiterplatte mit Prozeß 1 beginnen kann. ‚Bernstein' synchronisiert die Abarbeitung der Prozesse unabhängig von ihren Tasks. Die Abarbeitung der Prozesse erfolgt durch die spezifischen Geräteapplikationen. Daher ist auch ein autonomer Betriebsmodus der Geräteapplikationen (beispielsweise zu Testzwecken) ohne Beteiligung von ‚Bernstein' möglich.

Im Folgenden werden diejenigen Eigenschaften von ‚Evoscreen' und ‚Bernstein' aufgelistet, die diese Komponenten als Basis für die Implementierung des Testsystems qualifizieren:

- Das prozedurale Wissen für einen einzelnen Bearbeitungsschritt (Gerätekommando und Parameter) wird in einer einzelnen Aktivität (bzw. Task) definiert. Mehrere unmittelbar aufeinander folgende Aktivitäten, die durch dasselbe Gerät ausgeführt werden sollen, werden in einem ‚Evoscreen'-Prozeß gruppiert.

- Prozesse werden gemäß der Reihenfolge ihrer Ausführung in Prozeßsequenzen angeordnet.

- Die Zuordnung der Prozesse einer ‚Evoscreen'-Prozeßsequenz zu den ausführenden Geräten wird vorab definiert.

- ‚Bernstein' arbeitet unabhängig von Aktivitäten. Er kommuniziert lediglich mit Geräteapplikationen und synchronisiert die Ausführung der Prozesse.

- Die Geräteapplikationen laden ihre Prozeßdefinitionen aus einer Datenbank und arbeiten sie selbständig ab.

- Event-basiertes Scheduling erlaubt die Synchronisierung auch zeitlich unkalkulierbarer Prozesse.

---

[144] Die Evoscreen-Workflow-Definitionen enthalten die Zuordnung der Prozesse zu Laborgeräten. Diese Zuordnung gilt stets auch für alle Aktivitäten innerhalb eines Evoscreen-Prozesses. Daher kann eine Synchronisierung nur auf Prozess-Ebene erfolgen.

Wie in Kapitel 5.1 beschrieben, soll das Testsystem prozedurales Wissen[145] exakt dort zur Verfügung stellen, wo es vom Laborpersonal benötigt wird, und genau dann, wenn es genutzt werden soll. Das Testsystem soll also eine synchronisierte Verteilung prozeduralen Wissens an Laborpersonal leisten. Da ‚Bernstein' unabhängig vom Inhalt der Prozessdefinitionen arbeitet, läßt er sich grundsätzlich für diesen Zweck einsetzen. Um das Testsystem umzusetzen, wird eine Applikation benötigt, die auf Basis geeigneter Kommandos (beispielsweise zur Präsentation von Instruktionen an Laborpersonal) und entsprechenden Parametern mit dem Laborpersonal interagiert, mit ‚Bernstein' gemäß dessen Schnittstellenspezifikation kommuniziert und eine Synchronisierung durch ‚Bernstein' erlaubt. Diese Applikation muß sich also ‚Bernstein' gegenüber wie eine Geräteapplikation verhalten. Dann kann ‚Bernstein' ohne Änderungen verwendet werden, um mit mehreren Instanzen einer solchen Applikation zu interagieren. Eine solche Applikation wird im Folgenden als *HumanApplication* bezeichnet. Analog zu den gerätespezifischen Kommando- und Parametersätzen zum Steuern der einzelnen Laborgeräte wird ein Kommando- und Parametersatz für die Steuerung der HumanApplication benötigt. Die Kommandos müssen eine Interaktion mit dem Laborpersonal ermöglichen (wie etwa ‚InfoHTML' + Dateipfad zum Anzeigen prozeduralen Wissens im GUI der HumanApplication), und die HumanApplication muß diese Kommandos ausführen können.

Gemäß der Abstraktion in Kapitel 5.1 werden die einzelnen Aktivitäten der analysierten Laborprotokolle rein sequentiell abgearbeitet. Somit kann die oben beschriebene Struktur von ‚Evoscreen'-Workflow-Definitionen auch für die Repräsentation von Laborprotokollen verwendet werden. Jeder Bearbeitungsschritt eines biomedizinischen Laborprotokolls kann durch eine Workflow-Aktivität repräsentiert werden, die ein Kommando und Parameter an die HumanApplication zur Interaktion mit dem Anwender enthält (HumanTask). Mehrere solche Aktivitäten können in Form eines ‚Evoscreen'-Prozesses sequentiell gruppiert werden (HumanProcess).

Abhängig von Größe und Ausstattung eines biomedizinischen Labors ist seine Aufteilung in verschiedene *Laborstationen*[146] sinnvoll. Laborstationen können sich voneinander hinsichtlich der vorhandenen Laborgeräte unterscheiden. Da jede Laborstation das exakt benötigte prozedurale Wissen zur Verfügung stellen soll[147], ist die Struktur der ‚Evoscreen'-Workflow-Definitionen mit ihrer statischen Zuordnung von Prozessen zu Ressourcen vorteilhaft,

---

[145] gemäß den Ergebnissen der Abstraktion in 5.1.1 auch prozedurales Wissen zur Gerätebedienung (semimanuelle Präparationsschritte)

[146] Eine Laborstation ist ein Arbeitsplatz, am dem eine Instanz der HumanApplication ausgeführt wird.

[147] Die HumanApplication soll dem Laborpersonal auch Instruktionen für die Bedienung von Laborgeräten zur Verfügung stellen. Auf diese Weise können auch Geräte, für die keine eigene Geräteapplikation existiert, unter (mittelbarer) Synchronisierung durch den System Controller betrieben werden.

denn dies erlaubt auch die gezielte Zuordnung von HumanProcesses zu denjenigen Laborstationen, die über erforderliche Geräteausstattung verfügen. Somit kann eine gezielte Verteilung prozeduralen Wissens erfolgen. Einer der Labortrakte im Fraunhofer-IBMT wurde zum Beispiel in folgende Laborstationen unterteilt:

- Biochemische Manipulation
- Physikalische Manipulation
- Zellkultur
- Qualitätskontrolle
- Aliquotieren
- Einfrieren/Auftauen

Durch das beschriebene Konzept kann ein biomedizinisches Labor als Analogon eines HTS-Moduls betrachtet werden. Der Scheduler ‚Bernstein' kann zur Synchronisierung von HumanProcesses an verschiedenen Instanzen der HumanApplication verwendet werden. In Abhängigkeit des Kommando- und Parametersatzes der HumanApplication ist es möglich, prozedurales Wissen an das Laborpersonal zu verteilen oder Proben- oder Prozeßwissen vom Laborpersonal zu akquirieren. Diese Vorgänge sind durch Abarbeitung entsprechender Aktivitäten in der jeweiligen Workflow-Definition[148] realisierbar. Auf diese Weise können Personen als Aktoren in ein High-Throughput-fähiges Workflow-Management-System integriert werden. Umseitig abgebildete Grafik illustriert dies:

[148] Zum Erstellen der Workflow-Definitionen kann die Datenbankstruktur der Evoscreen-Workflow-Datenbank in unveränderter Form genutzt werden.

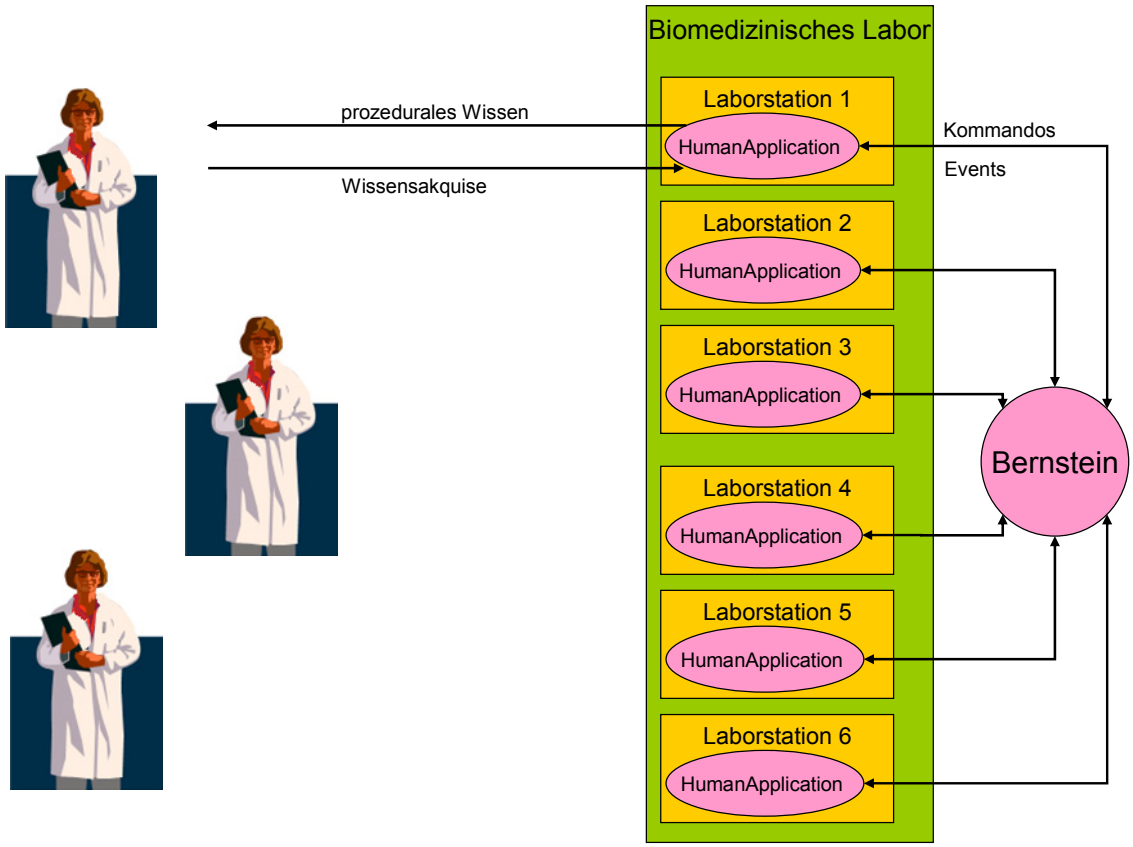

Abbildung 35: Konzeption des Testsystems. Es soll eine synchronisierte Verteilung prozeduralen Wissens an das Laborpersonal leisten, d.h. prozedurales Wissen zur Verfügung stellen, wo es benötigt wird, und zu den entsprechenden Zeitpunkten. ‚Bernstein' kann mit seinen existierenden Kommunikationsschnittstellen verwendet werden, um verschiedene HumanProcesses an mehreren Instanzen einer HumanApplication zu synchronisieren. Die Synchronisierung erfolgt auf Basis von Kommandos an die HumanApplication und entsprechenden Events an ein Kommunikationsthread von ‚Bernstein'. Die HumanApplication dient dem Präsentieren prozeduralen Wissens und der Akquise von Proben- und Prozeßwissen. Ihr Verhalten wird durch die Abarbeitung von HumanProcesses bestimmt, die aus HumanTasks bestehen. Ein HumanTask ist eine Workflow-Aktivität, die bei ihrer Abarbeitung eine Funktion der HumanApplication und entsprechende Parameter zur Interaktion mit dem Benutzer aufruft. Die Struktur der Workflow-Definitionen entspricht der Struktur von ‚Evoscreen'-Workflows. Eine Instanz der HumanApplication ist jeweils einer Laborstation zugeordnet, so dass ein Labor als Analogon zu einem automatischen Labormodul betrachtet werden kann, und Personen können auf diese Weise als Aktoren in ein High-Throughput-fähiges Workflow-Management-System integriert werden.

Im folgenden Abschnitt wird auf Basis dieses Konzeptes der erforderliche Kommando- und Parametersatz einer HumanApplication definiert und eine konkrete Implementierung der HumanApplication vorgestellt.

5.1.3 Implementierung der HumanApplication: ‚GenericDevice'

Um das oben beschriebene Konzept umzusetzen, wird eine HumanApplication benötigt, die sich ‚Bernstein' gegenüber wie eine Geräteapplikation verhält und mit dem Laborpersonal auf Basis von Workflow-Aktivitäten interagiert. Dazu erforderlich ist ein geeigneter Satz von

Kommandos und Kommandoparametern. Dieser Kommandosatz dient der Definition von Aktivitäten, die durch die HumanApplication abgearbeitet werden und ihr Verhalten steuern sollen. Er erweitert faktisch die Workflow-Beschreibungssprache von ‚Evoscreen' um die Möglichkeit, mit Laborpersonal zu interagieren. Benötigt werden Kommandos für:

- Ausgabe von Handlungsanweisungen an den Anwender
- Ausgabe von Alarmmeldungen
- Starten von Applikationen mit Parameterübergabe
- Akquise von Proben- und Prozesswissen mittels Formularen (siehe unten)

Bei der Entwicklung der konkreten HumanApplication ‚GenericDevice' wurden daher die folgenden Kommandos (*functions*) und Kommandoparameter implementiert:

- InfoText ([Description],[Text])
- InfoHTML ([Description],[Pfad/Dateiname])
- Alert ([Description],[Alarmtext])
- Start ([Description],[Pfad/Dateiname])
- ProcessForm([Formularname])

Grundsätzlich könnte das Präsentieren von Handlungsanweisungen an das Laborpersonal ausschließlich mit Hilfe von ‚InfoText'-Aktivitäten realisiert werden. Die jeweilige Handlungsanweisung kann in Form eines unformatierten Textes als Parameter der Aktivität definiert werden. Dabei gibt es jedoch folgende Schwierigkeiten: zum einen ist der ‚Evoscreen'-Prozesseditor nur für Parameter aus wenigen Zeichen konzipiert und wird unübersichtlich, wenn Texte in die Parameterfelder eingetragen werden. Zum anderen ist das Erscheinungsbild von unformatiertem Text im GUI des ‚GenericDevice' zur Vermittlung von Wissen denkbar ungeeignet. Insbesondere schwer explizierbares Handlungswissen, das in Form von Bildern oder Videos vermittelt werden könnte, läßt sich so nicht darstellen. ‚InfoHTML'-Aktivitäten hingegen erlauben eine beliebig formatierte Darstellung von Instruktionen an Laborpersonal in Form vorab entworfener HTML-Dateien. ‚InfoHTML'-Aktivitäten benötigen als Parameter den Namen und den Pfad der jeweils darzustellenden HTML-Datei. Bei Abarbeitung einer ‚InfoHTML'-Aktivität durch ‚GenericDevice' wird das ‚InfoHTML'-Kommando ausgeführt und die als Parameter übergebene HTML-Datei im GUI von ‚GenericDevice' dargestellt. Wie oben beschrieben, stehen beim Testsystem Aspekte wie Akzeptanz und Machbarkeit von Benutzerführung bei der Protokollabarbeitung durch ein Workflow-Managementsystem im Vordergrund, nicht die dezentrale Speicherung der Workflow-Definition bei der Probe. Daher ist die lokale Speicherung von HTML-Seiten auf Laborstationen oder auf einem zentralen Server für die Belange des Testsystems hinreichend.

Um die Akquise von Wissen durch die HumanApplication ‚GenericDevice' zu ermöglichen, wurden verschiedene Eingabeformulare implementiert, die durch Abarbeitung von ‚ProcessForm'-Aktivitäten durch ‚GenericDevice' aufgerufen und im GUI bearbeitet werden können[149]. Dadurch ist es möglich, die Ausgabe einer Handlungsanweisung bereits in der Workflow-Definition gezielt an ein Eingabeformular zu koppeln. In folgender Tabelle werden einige der in ‚GenericDevice' implementierten Formulare aufgelistet:

Formularname	**Zweck**	**Eingabefelder**
OKBox	einfache Vorgangs-bestätigung	keine
CPipetteForm	Dokumentation eines Pipettiervorgangs	Verwendete Menge, Reagenztemperatur, Bemerkungen
CIncubationForm	Erfassen von Inkubationsmerkmalen	Temperatur, Dauer, CO_2-Konzentration, Luftfeuchtigkeit
CAquireReagentData	Erfassen von Reagenzdaten	Hersteller, Herstellerbezeichnung, Charge, Seriennummer, Herstellungsdatum Konzentration
CRateOfVitality	Dokumentation der Vitalitätsrate	Vitalitätsrate (%)
CNumberOfCells	Dokumentieren einer Zellkonzentration	Zellzahl pro ml
CCentrifugation	Dokumentieren einer Zentrifugation	Volumen, Dauer, Bremse ein/aus, Temperatur, Gravitation
CAdjustDensity	Formular zur Berechnung und Dokumentation benötigter Medienmenge	Ursprungszellkonzentration, Ursprungsvolumen, absolute Zellzahl, gewünschte Zellkonzentration, Ergebnis

Tabelle 7: Auflistung einiger Formulare, die in der HumanApplication ‚Generic Device' zur Wissensakquise implementiert sind. Sie werden als Parameter des Kommandos ‚ProcessForm' aus Workflow-Aktivitäten heraus aufgerufen.

Auf Basis des implementierten Befehls- und Parametersatzes von ‚GenericDevice' können biomedizinische Laborprotokolle unter Verwendung der ‚Evoscreen'-Workflow-Struktur in Workflow-Definitionen überführt werden. Einzelne Protokollschritte werden dabei durch Aktivitäten repräsentiert, deren Kommandos und Parameter das ‚GenericDevice' zum Darstellen von entsprechenden Handlungsanweisungen (hinterlegt in Form von HTML-Dateien) oder von bestimmten Formularen veranlassen. Im Folgenden wird dies beispielhaft für einen Protokollschritt illustriert. Die unmittelbare anschließende Wissensakquise mittels zweier verschiedener Formulare wird ebenfalls in Form von Aktivitäten im Workflow definiert. Die Einbettung der beschriebenen Transkription in eine Prozeßdefinition ist weiter unten illustriert.

[149] Das weitere Vorgehen in Bezug auf akquiriertes Wissen wird in Kapitel 5.2 erläutert.

Protokollschritt	Aktivität	Beschreibung	Kommando	Parameter
„Pipettieren Sie 100µl Ethidiumbromid zu der Lösung hinzu“	Step 6	Ethidiumbromid hinzu	InfoHTML	929/1-3
	Step 7	Reagenzdaten erfassen	ProcessForm	CAcquireReagentData
	Step 8	Pipettierdaten erfassen	ProcessForm	CPipetteForm

Tabelle 8: Transkription eines Protokollschritts in eine Folge von Workflow-Aktivitäten unter Verwendung des Befehlssatzes und Parametersatzes von ‚GenericDevice’. Das prozedurale Wissen, das durch ‚GenericDevice’ dargestellt werden soll, wird lokal in Form einer HTML-Datei vorgehalten. Aktivität ‚Step 6’ führt bei ihrer Abarbeitung durch ‚GenericDevice’ zum Aufruf der Funktion ‚InfoHTML’ mit den Parametern ‚929’ als Pfadangabe und ‚1-3’ als Dateiname, so daß ‚GenericDevice’ die HTML-Datei im GUI anzeigt. Die Datenakquise wird durch die ‚ProcessForm’-Aktivitäten ‚Step 7’ und ‚Step 8’ im Workflow definiert. Als Parameter werden jeweils die Namen der darzustellenden Formulare verwendet. Abarbeitung dieser Aktivitäten führt zum Aufruf der Funktion ‚ProcessForm’ und zur Darstellung der anhand ihrer Namen definierten Formulare.

Die Definition der Aktivitäten in der ‚Evoscreen’-Datenbank und ihre Gruppierung zu ‚Evoscreen’-Prozessen wird unter Verwendung eines Editors durchgeführt. Wie oben dargelegt, können einzelne Aktivitäten derart zu Prozessen gruppiert werden, daß ihre Zuordnung zu einer geeigneten Laborstation gezielt definiert werden kann.

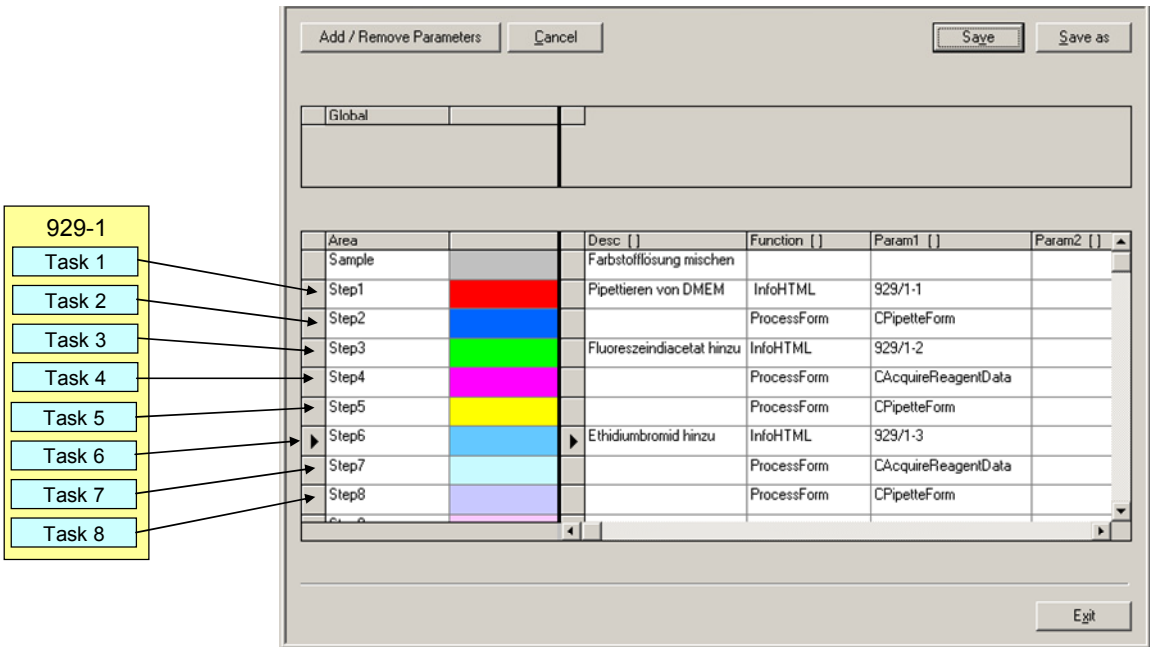

Abbildung 36: Links: schematische Darstellung des Prozesses ‚929-1’ in Anlehnung an Abbildung 34, bestehend aus acht HumanTasks. Rechts: der Prozeß-Editor von ‚Evoscreen’. Die einzelnen Tasks (hier als *Steps* bezeichnet) eines Prozesses werden mit kurzer Beschreibung, aufzurufendem Kommando und Kommandoparametern in der Workflow-Datenbank definiert. Die Steps 6, 7 und 8 in der Abbildung entsprechen den in Tabelle 8 beschrie-

benen Aktivitäten, die bei Abarbeitung durch ‚GenericDevice' sein Verhalten und damit die Interaktion mit dem Benutzer steuern.

Die resultierende Interaktion einer Instanz von ‚GenericDevice' mit dem Laborpersonal wird in der folgenden Abbildung seines GUI deutlich. Die oben beschriebene Aktivität ‚Step 6' steuert das Verhalten der Instanz des ‚GenericDevice':

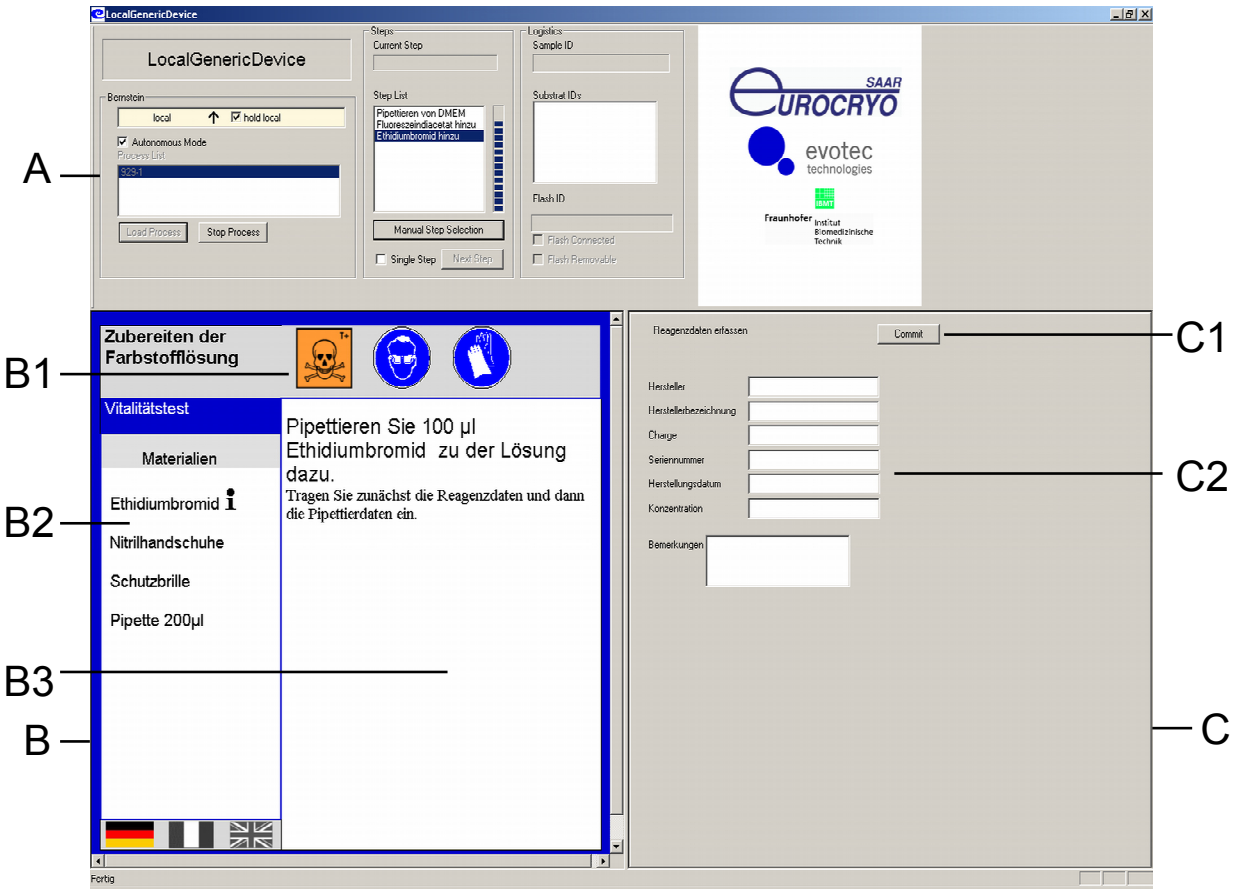

Abbildung 37: Interaktion des ‚GenericDevice' mit dem Anwender, basierend auf der Ausführung der beschriebenen Workflow-Aktivitäten ‚Step 6' und ‚Step 7'. (A) Statusbereich des ‚GenericDevice'. Hier werden die zur Abarbeitung durch die Station zugewiesenen Prozesse einer Workflow-Definition angezeigt, die einzelnen Tasks innerhalb des aktuellen Prozesses, ihr Bearbeitungsfortschritt anhand eines Fortschrittsbalkens und die IDs von Probe und Substrat. Verschiedene Bedienelemente ermöglichen autonomen Betriebsmodus oder das Laden einzelner Prozesse zu Testzwecken. (B) Ausgabebereich, in dem die per ‚InfoHTML'-Aktivität ‚Step 6' definierte HTML-Datei mit benötigtem prozeduralem Wissen dargestellt wird. (B1) Warnhinweise und Symbole zum Personenschutz, motiviert durch die Personengefährdung, die aus fehlerhaftem individuellem Wissen hervorgehen kann. (B2) Liste der im aktuellen Arbeitsschritt benötigten Materialien und Reagenzien. (B3) Bereich zur Vermittlung prozeduralen Wissen in Form von Handlungsanweisungen. (C) Eingabebereich bzw. Bereich für die Bearbeitung von Formularen. In der Abbildung aktiv ist das von Aktivität ‚Step 7' aufgerufene Formular zum Erfassen von Reagenzien. (C1) Schaltfläche zum Abschließen einer Formularbearbeitung (C2) Eingabefelder des Formulars.

Durch Implementierung der beschriebenen HumanApplication ‚GenericDevice' als Bindeglied zwischen dem Scheduler ‚Bernstein' und dem Laborpersonal wurde unter Verwendung der ‚Evoscreen'-Plattform und unter Beibehaltung der ‚Evoscreen'-Workflow-Struktur

ein erstes Testsystem für ein Workflow-Managementsystem umgesetzt, das eine Benutzerführung und einheitliche Datenakquise bei der Abarbeitung von Präparationsprotokollen in biomedizinischen Forschungslaboratorien leistet.

5.2 Eine erste Datenstruktur für dezentrale Wissensbewahrung

Das Konzept der Verteilung und Bewahrung deklarativen[150] und prozeduralen[151] probenbezogenen Wissens durch Kopplung von dezentralen elektronischen Speicherchips und biologischen Proben erfordert im Kontext von Biobanking die Kryotoleranz solcher Speichermedien.

Experimentelle Untersuchungen [Ihmig 2006] haben die Eignung bestimmter Compact-Flash Speicherkarten gezeigt. Dadurch wurde es erstmals möglich, prototypische funktionale Einheiten (im Folgenden als *Unity* bezeichnet) aus Substraten und kryotoleranten elektronischen Speichermedien zu konstruieren. Das Design der erforderlichen Probenträger wurde dabei maßgeblich vom Compact-Flash-Formfaktor beeinflußt, ebenso vom ursprünglichen Ziel der Miniaturisierung von Probenvolumina[152]. Verschiedene solcher Prototypen werden in Kapitel 4.2 beschrieben, unter anderem ein Schiebesubstrat mit 30 Probentöpfchen und daraus zusamengesteckte Substratstapel. Die darauf aufbauende prototypische Probenschublade zur elektronischen Konnektierung und Organisation von Substratstapeln wird ebenfalls in Kapitel 4.2 gezeigt.

Vor dem Hintergrund dieser Entwicklungen war erstmals ein technologisch fundierter Rahmen für die Entwicklung von Datenstrukturen für dezentrale Wissensbewahrung gegeben, aus dem sich gleichzeitig Anforderungen an die Datenstruktur ergeben haben. Als Rahmenbedingungen sind beispielsweise die Performanz von Speichermedien unter kryogenen Temperaturen bei Lese- und Schreibzugriffen zu nennen [Ihmig 2006], als Anforderungen die Datenfelder zur Abbildung und Beschreibung der Unity, ihrer Komponenten, der enthaltenen Proben, des entsprechenden deklarativen sowie prozeduralen Wissens und der Labordokumentation. In den folgenden Abschnitten werden Anforderungen aufgezeigt und die Konzeption einer ersten Datenstruktur für dezentrale Wissensbewahrung beschrieben.

---

[150] Siehe die in Kapitel 2.1 aufgelisteten Wissensarten im Biobanking.

[151] Prozedurales Wissen in Form einer WMS-kompatiblen Workflow-Definition

[152] Eine Miniaturisierung von Probenvolumina wurde von den biophysikalischen wie auch logistischen Vorteilen kleiner Volumina motiviert. Siehe auch Kapitel 4.2.

5.2.1 Grundsätzliche Anforderungen

In [Durst 2003] wurde nachgewiesen, daß eine wesentliche Schwierigkeit beim Einsatz von Datenbanken zur Bewahrung deklarativen Probenwissens die hohe Dynamik der Informationsanforderungen ist. Die Informationsanforderungen werden beispielsweise durch biowissenschaftlichen Fortschritt, technologische Weiterentwicklung (beispielsweise von Probenträgern, elektronischen Speicherchips und Lagertechnologie) oder durch Erweiterung des Lagerspektrums von Biobanken um weitere Probentypen[153] kontinuierlich verändert. Eine Datenstruktur, die als Basis für dezentrales Bewahren probenrelevanter Information dienen soll, unterliegt ebensolchen Veränderungen und Erweiterungen und muß zum Zeitpunkt des Einlagerns einer Unity den aktuellen Informationsanforderungen entsprechen. Während des Betriebs von Biobanken führt die kontinuierliche Weiterentwicklung der Datenstruktur einerseits zu immer neuen, aktuellen Versionen und andererseits zur Veraltung einer Vielzahl unterschiedlicher Versionen, die mit entsprechendem Probenwissen gefüllt und an Proben angekoppelt unter Kryobedingungen lagern.

Aus den geschilderten Sachverhalten ergeben sich zwei grundsätzliche Anforderungen an die Datenstruktur:

- Die Datenstruktur für dezentrale Wissensbewahrung (und –verteilung) muß einfach und zeitnah erweiterbar und an neue Informationsanforderungen anzupassen sein
- Die ‚alten' Versionen der Datenstruktur müssen langzeitkompatibel sein

Hinsichtlich der wünschenswerten Unabhängigkeit der Datenstruktur von proprietären Dateiformaten kommerzieller Anbieter ist ein ‚zeitloses', offenes Format erforderlich, das auch bei langer Lagerzeit die Lesbarkeit/Nutzbarkeit der Daten gewährleistet und somit eine grundsätzliche Kompatibilität sichert. Im Hinblick auf die Kompatibilität mit einer Probendatenbank und deren jeweils aktuellem Datenmodell sind versionsbasierte Filter oder Konverter erforderlich, damit die ‚veralteten' Versionen der Datenstruktur dargestellt und ausgewertet oder konvertiert werden können.

Datenstrukturen, die den formulierten Anforderungen genügen, lassen sich unter Verwendung der *Extensible Markup Language* (*XML*) definieren. XML ist eine Beschreibungssprache, die frei erweiterbar ist und es erlaubt, benötigte *Elementtypen* selbst zu definieren, um ein Dokument hierarchisch und in logische Komponenten (*Elemente*) strukturiert abzubilden. Ein XML-Dokument ist definiert als eine Folge aus Markup und Zeichendaten und kann

---

[153] Eine konkrete derartige Erweiterung fand in der Eurocryo-Biobank im Zuge des GHRC-Projektes statt. Umfangreiche Änderungen und Erweiterungen an der bestehenden Probendatenbank waren erforderlich, um die Eigenschaften von HIV-infizierten Zellproben und assoziierten Proben abbilden zu können.

unter Verwendung von XML-Prozessoren (*XML-Parser*) durch Applikationen verarbeitet werden. XML ist mit Standard-Texteditoren kompatibel und kann mit ihnen erzeugt, bearbeitet und gelesen werden[154]. Dadurch sind die Daten in XML-Dokumenten frei zugänglich, und die Unabhängigkeit von Hersteller-proprietären Datenformaten ist gewährleistet. Die Darstellung von XML-Dokumenten (beispielsweise ‚veralteter' Datenstrukturen) kann durch für die jeweilige Version des Dokuments geeignete Stylesheets erfolgen, und die Konvertierung in eine neuere Datenstruktur mit Hilfe entsprechender XML-Prozessoren und Konversionsdefinitionen stattfinden.

5.2.2 Konzeption einer ersten Datenstruktur für dezentrale Wissensbewahrung

Für die Konzeption einer ersten Datenstruktur zur dezentralen Wissensbewahrung unter kryogenen Bedingungen werden folgende Voraussetzungen angenommen: eine Unity besteht aus einem Compact-Flash und maximal fünf aufgesteckten Schiebesubstraten[155] mit jeweils maximal 30 Einzelwells[156]. Es wird angenommen, daß alle Substrate Aliquoten derselben Probe enthalten[157]. Ferner wird vorausgesetzt, daß alle Substrate einer Unity demselben Präparationsprotokoll unterzogen werden sollen.

Die Informationsanforderungen an die Datenstruktur ergeben sich dann im Wesentlichen aus den Bestandteilen der Unity (Compact-Flash und Substrate), der Unity selbst, der in den Wells der Substrate enthaltenen Probe und aus der Integration der Unity in das (in Kapitel 5.1 beschriebene) Testsystem. Die Datenstruktur läßt sich somit strukturieren in relevantes Wissen über:

- die Unity selbst
- die Einzelkomponenten der Unity:
    - ein Compact-Flash
    - maximal fünf Schiebesubstrate

---

[154] Dies resultiert aus der Kompatibilität von 7-bit-ASCII und den ersten 128 Zeichen von Unicode, dem Zeichensatz von XML.

[155] Die Anzahl der Substrate ist durch die Aufteilung der prototypischen Schublade limitiert.

[156] Dies entspricht den zu diesem Zeitpunkt real verfügbaren Substrattypen, siehe Kapitel 4.2.

[157] Diese Vereinfachung gegenüber den weiteren möglichen Varianten (Beispiel: alle Wells können unterschiedliche Proben enthalten) entspricht der gängigen Verwendung der Unities. Außerdem wird dadurch das vielfach redundante Speichern identischer Probendaten (für jedes einzelne Well jedes Substrates = 150 redundante Datensätze) vermieden.

- die Probe
- das der Unity zugeordnete Präparationsprotokoll in Form einer ‚Evoscreen'-Workflow-Definition (Run, siehe die Beschreibung des Testsystems in Kapitel 5.1)
- die Labordokumentation (Prozessdokumentation und akquiriertes Probenwissen)

Im Folgenden werden XML-Elementtypdefinitionen verwendet, um die konzipierte Datenstruktur und ihre Komponenten zu beschreiben. Gemäß dem oben genannten relevanten Wissen läßt sich das *Root*-Element der Datenstruktur wie folgt konzipieren:

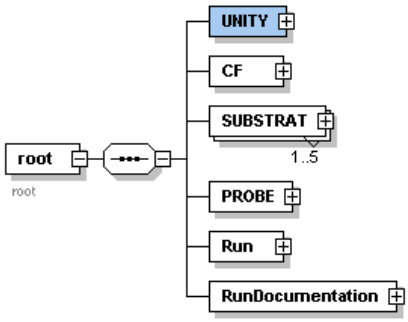

Abbildung 38: Das Root-Element einer ersten Datenstruktur für dezentrale Wissensbewahrung. Die Datenstruktur muß Wissen über die Unity, den enthaltenen Compact-Flash, ein oder mehrere (maximal fünf) Substrate und über die Probe repräsentieren können. Die Integration der Unity in das Testsystem (siehe auch Kapitel 5.1) erfordert die zusätzlichen Elementtypen ‚Run' und ‚Rundocumentation'.

In den folgenden Abschnitten werden die abgebildeten Elementtypen mit ihrer jeweiligen internen Struktur beschrieben.

5.2.2.1 Elementtyp ‚Unity'

Der Elementtyp ‚Unity' soll relevantes Wissens über die Unity selbst bewahren. Bei der Konzeption der ersten Datenstruktur wurden Unity ID, Datum der Zusammenkopplung von Substraten und CompactFlash sowie Lagerort der Unity als relevantes Wissen identifiziert. Das Wissen über den Lagerort ist hinsichtlich der Lokalisierung von Proben während ihrer Kryokonservierung bedeutsam. Dies setzt eine Konnektierung der Datenträger während ihrer Lagerung voraus[158]. Nachfolgende Abbildung zeigt schematisch den Aufbau des Elementtyps ‚Unity':

[158] Die Konnektierung während der Kryokonservierung ist für lesende und schreibende Zugriffe auf die Datenstruktur erforderlich, aber auch für eine exakte Lokalisierung. Eine solche Konnektierung war bereits zu dem damaligen Zeitpunkt geplant, wurde aber erst durch Realisierung der in Kapitel 4.4 beschriebenen elektronischen Tank-Infrastruktur umgesetzt.

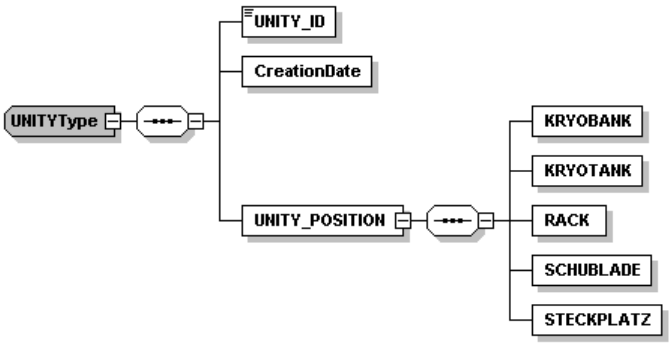

Abbildung 39: Elementtyp ‚Unity'.

5.2.2.2 Elementtyp 'CF' (Compact-Flash)

Als relevantes Wissen über den dezentralen Speicherchip wurden seine ID und weitere spezifische Informationen wie beispielsweise Hersteller, Modell, Revision und Speicherkapazität identifiziert. Dieses Wissen kann beispielsweise erforderlich sein, um optimierte Schreib-/Lesevorgänge unter kryogenen Bedingungen durchzuführen, da sich die Performanz zwischen verschiedenen Compact-Flash-Speichern unterschiedlicher Hersteller und unterschiedlicher Speicherkapazität voneinander unterscheiden kann.

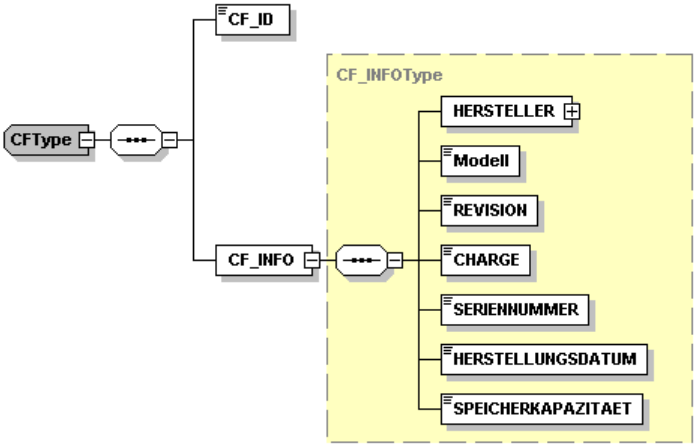

Abbildung 40: Elementtyp ‚CF'

5.2.2.3 Elementtyp ‚Substrat'

Bis zu fünf Schiebesubstrate können in einer Unity enthalten sein. Es wurde (siehe oben) vorausgesetzt, daß jedes Substrat maximal 30 einzelne Probentöpchen (*Einzelwells*) enthalten kann, und daß in jedem Einzelwell Aliquoten derselben Probe enthalten sind. Relevantes Wissen über ein Substrat sind dann unter anderem die tatsächlich vorhandene Anzahl der Einzelwells, deren Volumen, aber auch Angaben zum jeweiligen Substrattyp, der Revision, Chargennummer und Hersteller. Auf der Ebene der Einzelwells ist relevant, welches Einzelwell tatsächlich mit der Probe befüllt ist, damit ein Überblick über die Anzahl der vorhandenen Aliquoten und damit eine Inventarverwaltung möglich ist.

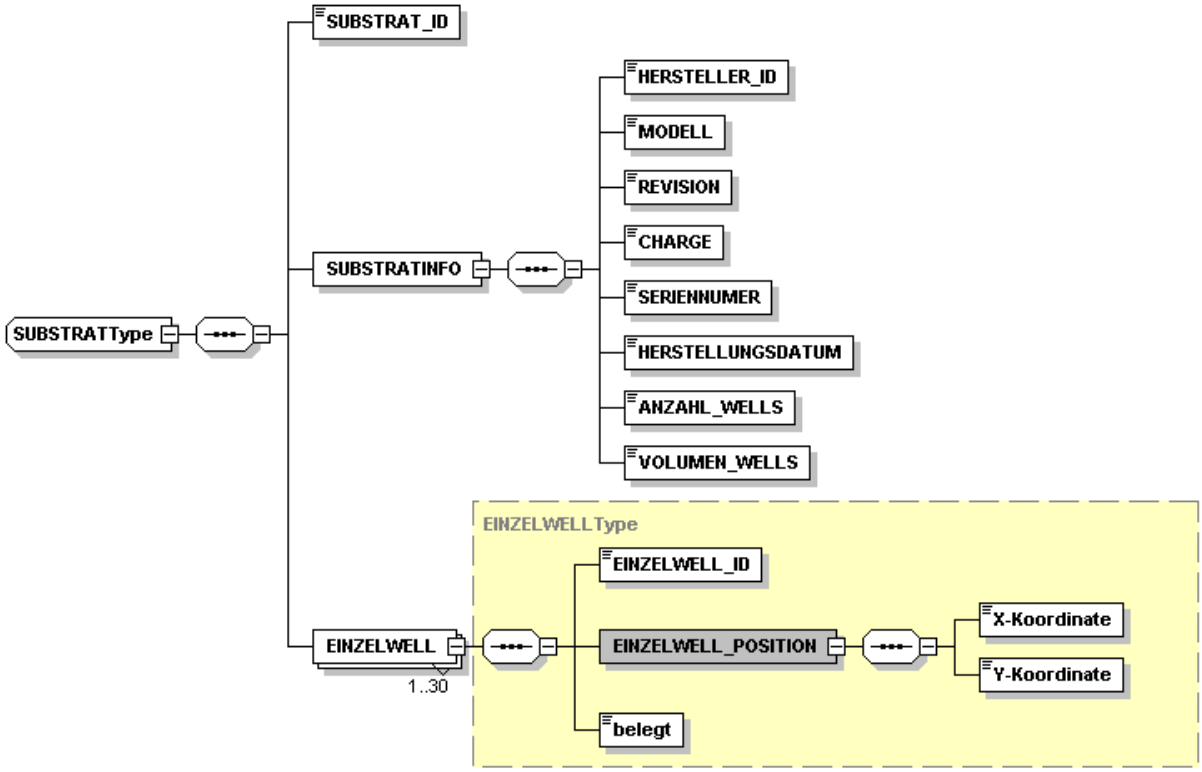

Abbildung 41: Elementtyp ‚Substrat'. Unterschiedliche Proben in den Einzelwells würden eine Erweiterung des Elementtyps ‚Einzelwell' erfordern. In diesem Fall müßte der in 5.2.2.4 beschriebene Elementtyp ‚Probe' in den Elementtyp ‚Einzelwell' integriert und aus dem Elementtyp ‚root' entfernt werden. Würde man annehmen, daß die Substrate einer Unity jeweils unterschiedlichen (auf Einzelwellebene aber einheitlichen) Präparationen unterzogen werden dürfen, müßte der Elementtyp ‚Substrat' um die in 5.2.2.5 und 5.2.2.6 beschriebenen Elementtypen ‚Run' und ‚Rundocumentation' erweitert und der Elementtyp ‚root' um diese beiden Elementtypen reduziert werden.

5.2.2.4 Elementtyp ‚Probe'

Wissen zu biologischen Proben ist teilweise sehr umfangreich und extrem unterschiedlich. Relevant zur Charakterisierung und Einordnung einer Probe ist jedoch ihr Zelltyp, aus dem sich bereits viele Merkmale und Eigenschaften implizit ergeben. Beispielsweise läßt sich die Herkunft einer Probe (beispielsweise human, tierisch oder pflanzlich) aus dem Zelltyp erkennen. Eine davon unabhängige Beschreibung wie auch das dezentrale Speichern beliebiger und beliebig vieler Merkmale ist erforderlich. Im Fall von humanen Proben kann der Vermerk einer PatientenID erforderlich sein, die auf weitere Informationen zum Patienten verweist, die in der Datenstruktur aus datenschutzrechtlichen Gründen nicht als Standard-Text abgelegt werden können.

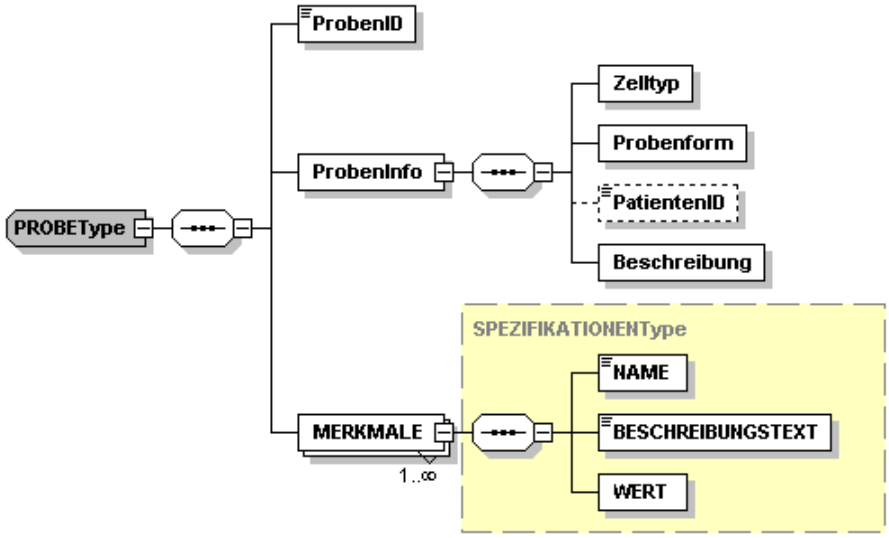

Abbildung 42: Elementtyp ‚Probe'. Relevante Probeneigenschaften können mit Hilfe beliebig vieler selbst definierbarer Merkmale definiert und in der Datenstruktur gespeichert werden. Zelltyp und Beschreibung erlauben bereits eine weitgehende Einordnung der Probe.

5.2.2.5 Elementtyp ‚Run'

Präparationsprotokolle können für die Abarbeitung durch das in Kapitel 5.1 beschriebene Testsystem als Runs definiert werden. Dies erfolgt unter Verwendung des Befehls- und Parametersatzes von ‚Generic Device' (siehe Abschnitt 5.1.3), unter Beibehaltung der Struktur von ‚Evoscreen'-Workflow-Definitionen und unter Verwendung der ‚Evoscreen'-Datenbank (siehe Abschnitt 5.1.2). Die Handlungsanweisungen für das Laborpersonal sind in Form einzelner, zentral gespeicherter HTML-Dateien für jeden einzelnen Präparationsschritt definiert und können durch Abarbeitung der einzelnen Workflow-Aktivitäten durch ‚Generic Device' aufgerufen und im GUI präsentiert werden (siehe auch Abschnitt 5.1.3). Die Architektur von ‚Bernstein' und ‚Evoscreen' ist ausschließlich auf Verwendung der ‚Evoscreen'-Datenbank ausgerichtet[159]. Daher ist eine (wie in Kapitel 4.5 konzipierte) vollständige dezentrale Speicherung prozeduralen Präparationswissens bei der Probe nicht mit dem implementierten Testsystem durchführbar. Hinsichtlich der Intention des Testsystems kann jedoch auf eine vollständige dezentrale Speicherung der Workflow-Definitionen verzichtet werden. Um unter diesen Umständen dennoch das Prinzip *Probe steuert ihre Präparation* realisieren zu können, muß das Wissen über die auszuführende Workflow-Definition in Form einer Referenz auf eine Workflow-Definition in der ‚Evoscreen'-Datenbank (bestehend aus Run-Name und Run-ID) in der Datenstruktur gespeichert werden können. Eine erfolgte Abarbeitung wird durch einen booleschen Wert in der Datenstruktur

[159] Eine Modifikation der Evoscreen-Software, der Datenbank oder des Schedulers ‚Bernstein' zur Verwendung XML-basierter dezentraler Workflow-Definitionsstrukturen war im Rahmen der vorliegenden Arbeit aus rechtlichen Gründen nicht möglich.

markiert, so daß dieser Run nicht mehrmals ausgeführt wird. Eine solche Referenz und Markierung sind im Elementtyp ‚Run' enthalten:

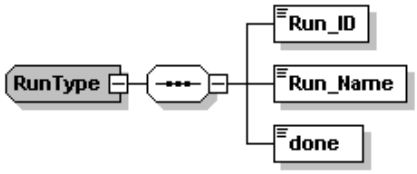

Abbildung 43: Elementtyp ‚Run' zur Integration der dezentralen Datenstruktur in das Testsystem. Eine Referenz auf die ‚Evoscreen'-Datenbank ermöglicht die Simulation probengesteuerter Präparation und die grundsätzliche Beurteilung des Konzepts (siehe auch Kapitel 5.1). Ein booelescher Wert verhindert die irrtümliche Mehrmals-Ausführung des ausgewählten Runs.

5.2.2.6 Elementtyp ‚Rundocumentation'

Die Abarbeitung einer Workflow-Definition kann unter Verwendung des Elementtyps ‚Run' (Abschnitt 5.2.2.5) quasi durch die Probe initiiert und gesteuert werden. Die Abarbeitung selbst erfolgt durch eine oder mehrere Instanzen der Applikation ‚Generic Device' (siehe Abschnitt 5.1.3). Die Workflow-Definitionen bestehen aus einzelnen Prozessen, die wiederum aus verschiedenen Aktivitäten auf Basis des Befehlssatzes von ‚Generic Device' (siehe Abschnitt 5.1.3) aufgebaut sind. Diese Aktivitäten steuern das Verhalten von ‚Generic Device' und damit die Interaktion mit dem Laborpersonal. Der Befehl ‚InfoHTML' etwa präsentiert beispielsweise eine formatierte Ausgabe, während der Befehl ‚ProcessForm' verschiedene Eingabeformulare für die Wissensakquise aufruft und verarbeitet. Die Verarbeitung des akquirierten Wissens umfaßt auch die Interaktion mit dem Elementtyp ‚RunDocumentation' der Datenstruktur für dezentrale Wissensbewahrung. Eine einheitlich aufgebaute dezentrale Dokumentation ist das Resultat dieser Interaktion. Der Elementtyp ‚RunDocumentation' wird im Folgenden beschrieben.

An den verschiedenen Laborstationen werden stets vordefinierte Gruppen von Aktivitäten (‚HumanProcesses') abgearbeitet. Die Dokumentation eines Workflows soll modular aus den Dokumentationen zusammengesetzt werden, die an den einzelnen Laborstationen anfallen, d.h. sie soll aus den Dokumentationen der einzelnen Prozesse aufgebaut sein. Ein solcher modularer Aufbau der Workflow-Dokumentation kann durch den Elementtypen ‚Rundocumentation' erreicht werden:

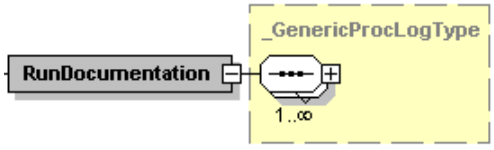

Abbildung 44: Der Elementtyp ‚RunDocumentation'. Die Dokumentation der Einzelprozesse eines Runs erfolgt modular durch Elemente des Typs ‚_GenericProcLog'.

Folgende Anforderungen wurden an die Dokumentation der einzelnen Prozesse eines Workflows (und damit an den Elementtypen ‚_GenericProcLog') definiert:

(a) Jeder Prozeß soll anhand der IDs der präparierten Substrate (maximal fünf, siehe oben) und dem Startzeitpunkt des Prozesses eindeutig identifiziert werden können.

(b) Prozeßname und Name der ausführenden Laborstation sollen dokumentiert werden.

(c) Die dezentrale Dokumentation der Prozesse muß von der künftigen Verfügbarkeit der Definitionsdatenbank und von eventuellen Änderungen an den Workflow- oder Prozessdefinitionen unabhängig sein. Daher ist die Beschreibung der einzelnen Aktivitäten mit ihrer jeweiligen Bezeichnung, ihrem Befehl und ihren Befehlsparametern in der Dokumentation notwendig.

(d) Für jeden Prozeß sollen Bemerkungen des Laborpersonals und die für jede Aktivität akquirierten Dokumentationsdaten abgelegt werden können. Für jede Aktivität sollen der Name des verwendeten Eingabeformulars, die Namen der Eingabefelder und die vom Laborpersonal eingegebenen Werte dokumentiert werden. Außerdem sollen die Zeitpunkte für Prozeßbeginn, Prozeßende, Aktivitätsbeginn und Aktivitätsende aufgezeichnet werden.

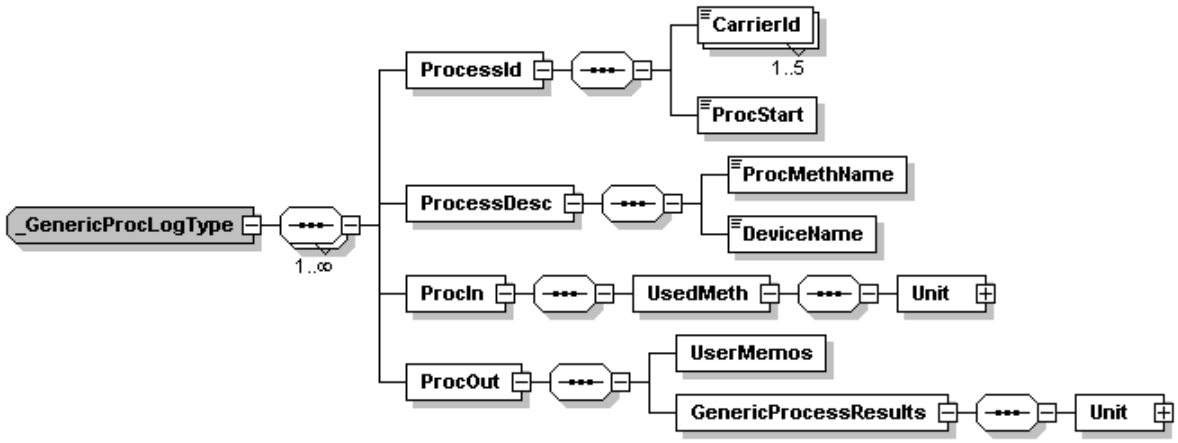

Abbildung 45: Der Elementtyp ‚_GenericProcLog' zum Dokumentieren der Einzelprozesse eines Runs. Anforderung (a) wird durch den Elementtyp ‚ProcessID' erfüllt, Anforderung (b) durch den Elementtyp ‚ProcessDesc'. Anforderung (c) und Anforderung (d) entsprechen den Elementtypen ‚ProcIn' bzw. ‚ProcOut' unter Verwendung des wiederverwendbaren Elementtyps ‚Unit' zur Dokumentation der Aktivitäten (siehe unten).

Es wurde ein wiederverwendbarer Elementtyp (‚Unit', siehe unten) entworfen, der sowohl für die Beschreibung der einzelnen Aktivitäten (‚Steps' beim Elementtyp ‚ProcIn') als auch für die Dokumentation ihrer Ergebnisse (beim Elementtyp ‚ProcOut') genutzt werden kann. Der Elementtyp ‚Unit' dient also allgemein zur Dokumentation von Aktivitäten:

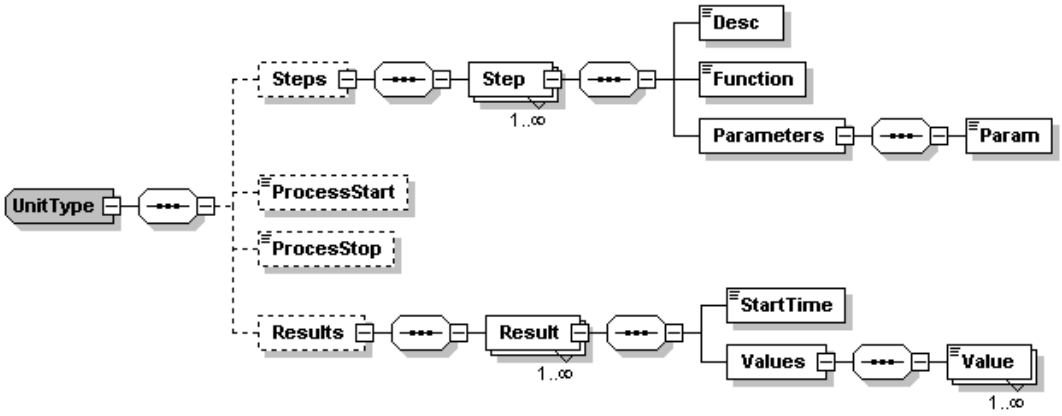

Abbildung 46: wiederverwendbarer Elementtyp ‚Unit'. Das obere Element, beginnend beim optionalen Element ‚Steps', wird im Zusammenhang mit dem Elementtyp ‚ProcIn' (siehe oben) für die Beschreibung der Aktivitäten (Beschreibung, Befehl, Parameter, beispielsweise ‚Pipettieren', ‚InfoHTML', ‚929-1') eines Prozesses verwendet. Die optionalen Elemente ‚ProcessStart' und ‚ProcessStop' werden zum Dokumentieren der Start- und Stopzeit des gesamten Prozesses genutzt. Das optionale Element ‚Results' dient zum Dokumentieren der Ergebnisse der Aktivitäten eines Prozesses. Jeder einzelne Wert eines verwendeten Eingabeformulars wird durch jeweils ein Value-Element, das Namen und Wert des Eingabefeldes enthält, dokumentiert.

5.2.3 Integration und Interaktion von Datenstruktur und Testsystem

Die Datenstruktur aus Kapitel 5.2 wurde für die Verwendung mit dem in Kapitel 5.1 beschriebenen Testsystem konzipiert. Eine Interaktion erfordert zusätzlich:

- Anpassungen am Scheduler ‚Bernstein'
- Anpassungen an der Applikation ‚Generic Device'

‚Bernstein' wurde dahingehend angepaßt, ein angeschlossenes Compact-Flash-Lesegerät kontinuierlich auf das Vorhandensein einer Instanz der in Abschnitt 5.2.2 konzipierten Datenstruktur zu prüfen, gegebenenfalls aus dem Element ‚root/Run/RunID' und ‚root/Run/Run_Name' eine Referenz auf die entsprechende Workflow-Definition in der ‚Evoscreen'-Datenbank zu lesen, die Workflow-Definition zu laden, die darin enthaltenen Prozessreferenzen an die zugeordneten Instanzen von ‚GenericDevice' zu verteilen und anschließend die Synchronisierung der Prozesse zu beginnen. Durch die so realisierte Integration der Datenstruktur werden die manuellen Schritte zur Auswahl und zum Laden einer Workflow-Definition und zum Starten der Synchronisierung überflüssig. Stattdessen steuert die Probe anhand der Informationen in der Datenstruktur den Start der Workflow-Abarbeitung. Dies entspricht im Wesentlichen dem Prinzip *Probe steuert ihre eigene Präparation* aus Kapitel 4.5.

Eine Prüfung auf Vorhandensein der Datenstruktur wurde auch in ‚Generic Device' integriert. Zwar erfolgt die Synchronisierung der Prozesse an den einzelnen Laborstationen uneingeschränkt durch ‚Bernstein', jedoch ist die Voraussetzung für den Start eines Prozesses durch eine Instanz von ‚Generic Device' das Vorhandensein eines Compact-Flash mit einer

Instanz der Datenstruktur. Damit ist gesichert, daß die Dokumentation der Prozesse direkt auf den Chip geschrieben werden kann[160]. Die Dokumentation der einzelnen Aktivitäten erfolgt anhand des in Abschnitt 5.2.2.6 definierten Elementtyps ‚Unit'. Dazu zeichnet ‚Generic Device' einerseits für jede Aktivität den enthaltenen Befehl und seine Parameter auf und dokumentiert andererseits die im zugeordneten Formular akquirierten Benutzereingaben. Eine exemplarische Dokumentation, die auf diese Weise entstanden ist, wird in folgender Abbildung auszugsweise gezeigt:

_GenericProcLog (25)

	= xmlns:xsd	= xmlns:xsi	ProcessId	ProcessDesc	ProcIn	ProcOut
1	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
2	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
3	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
4	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
5	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
6	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
7	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
8	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
9	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
10	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instance	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...
11	http://www.w3.org/2001/XMLSchema	http://www.w3.org/2001/XMLSchema-instanc	ProcessId	ProcessDesc	ProcIn	ProcOut xsi:type...

Abbildung 47: auszugsweise Darstellung einer Abarbeitung von 25 Prozessen, akkumuliert in der Datenstruktur für dezentrale Wissensbewahrung durch Instanzen von ‚Generic Device'. Für jeden abgearbeiteten Prozeß wurde innerhalb der Datenstruktur im Elementtyp ‚RunDocumentation' eine neue Instanz des Elementtyps ‚_GenericProcLog' angefügt. Jede Instanz besteht aus den Elementen ‚ProcessID', ‚ProcessDesc', ‚ProcIn' und ‚ProcOut', deren Strukturen den in Abschnitt 5.2.2.6 beschriebenen Elementtypdefinitionen entsprechen. ‚_GenericProcLog' dokumentiert für jeweils einen Prozeß die IDs der präparierten Substrate (siehe Abbildung 48), den Prozeßnamen und die zugeordnete Laborstation (siehe Abbildung 49), eine Beschreibung des Prozesses in Form der Auflistung seiner Aktivitäten mit den verwendeten Parametern (siehe Abbildung 50) und die akquirierten Werte aus den Eingabeformularen (siehe Abbildung 51). Zusätzlich zur dezentralen Wissensbewahrung auf dem Speicherchip wird auf einem als *Online-DB* bezeichneten gemeinsamen Netzlaufwerk für jeden abgearbeiteten Prozeß jeweils eine separate XML-Datei erzeugt, die nur aus einem einzigen ‚GenericProcLog'-Element besteht.

[160] Das Vorhandensein eines Compact-Flash-Speichers wird im GUI von ‚Generic Device' angezeigt. Gleiches gilt für seine Entnehmbarkeit nach erfolgten Schreibzugriffen in die Datenstruktur.

Drei exemplarische Elemente vom Typ ‚ProcessID' aus der in Abbildung 47 beschriebenen Dokumentation sind im Folgenden auszugsweise gezeigt:

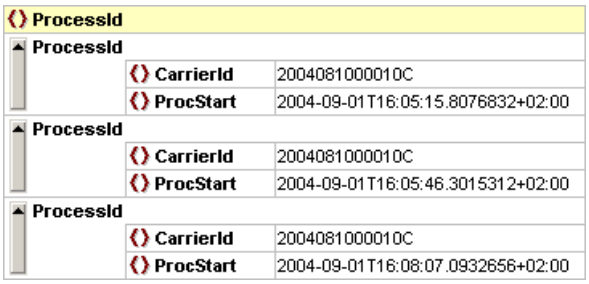

Abbildung 48: Drei Elemente vom Typ ‚ProcessID' aus Abbildung 47 im Detail. Der Elementtyp ‚ProcessID' dokumentiert innerhalb des Elementtyps ‚_GenericProcLog' die IDs der in einem Prozeß präparierten Substrate und den Startzeitpunkt des jeweiligen Prozesses.

Die zu den in Abbildung 48 gehörenden Prozeßbeschreibungen werden in folgender Abbildung anhand der Elemente vom Typ ‚ProcessDesc' gezeigt:

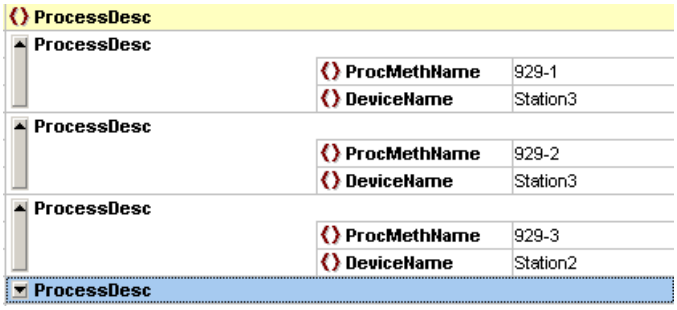

Abbildung 49: Die Prozessbeschreibungen der drei in Abbildung 48 exemplarisch gezeigten Prozesse. Der Elementtyp ‚ProcessDesc' dokumentiert innerhalb des Elementtyps ‚_GenericProcLog' den Namen des jeweiligen Prozesses und die per Rundefinition zugeordnete Laborstation.

Folgende Abbildung zeigt exemplarisch die Beschreibung der beiden Aktivitäten (Steps) des Laborstation 3 zugeordneten Prozesses ‚929-2':

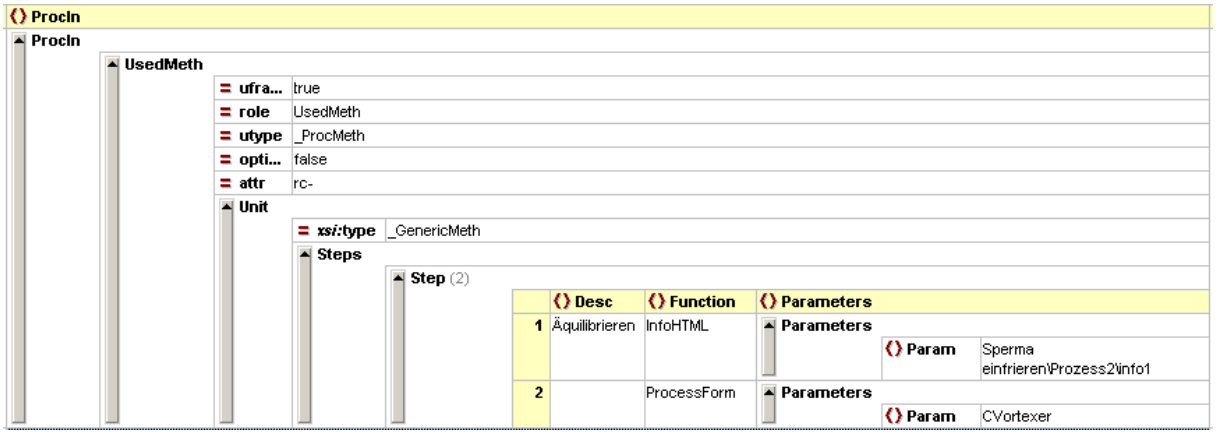

Abbildung 50: Beschreibung der beiden Aktivitäten des Prozesses ‚929-2' in der Dokumentation anhand des Elementtyps ‚ProcIn'. Für jede Aktivität (Step) des Prozesses werden unter anderem die Beschreibung (‚Desc'), der aufgerufene Befehl (‚Function') und seine Parameter (‚Parameters') aufgezeichnet. Diese Struktur entspricht der in Abbildung 36 gezeigten Definition von Schritten innerhalb eines Prozesses.

Die durch Verarbeitung von Eingabeformularen akquirierte Dokumentation von Aktivitäten und die automatisch erfassten Zeitstempel werden unter Verwendung von Instanzen des Elementtyps ‚ProcOut' gespeichert. Folgende Abbildung zeigt dies auszugsweise für einen Teil der in Abbildung 47 im Überblick dargestellten Dokumentation von 25 Prozessen:

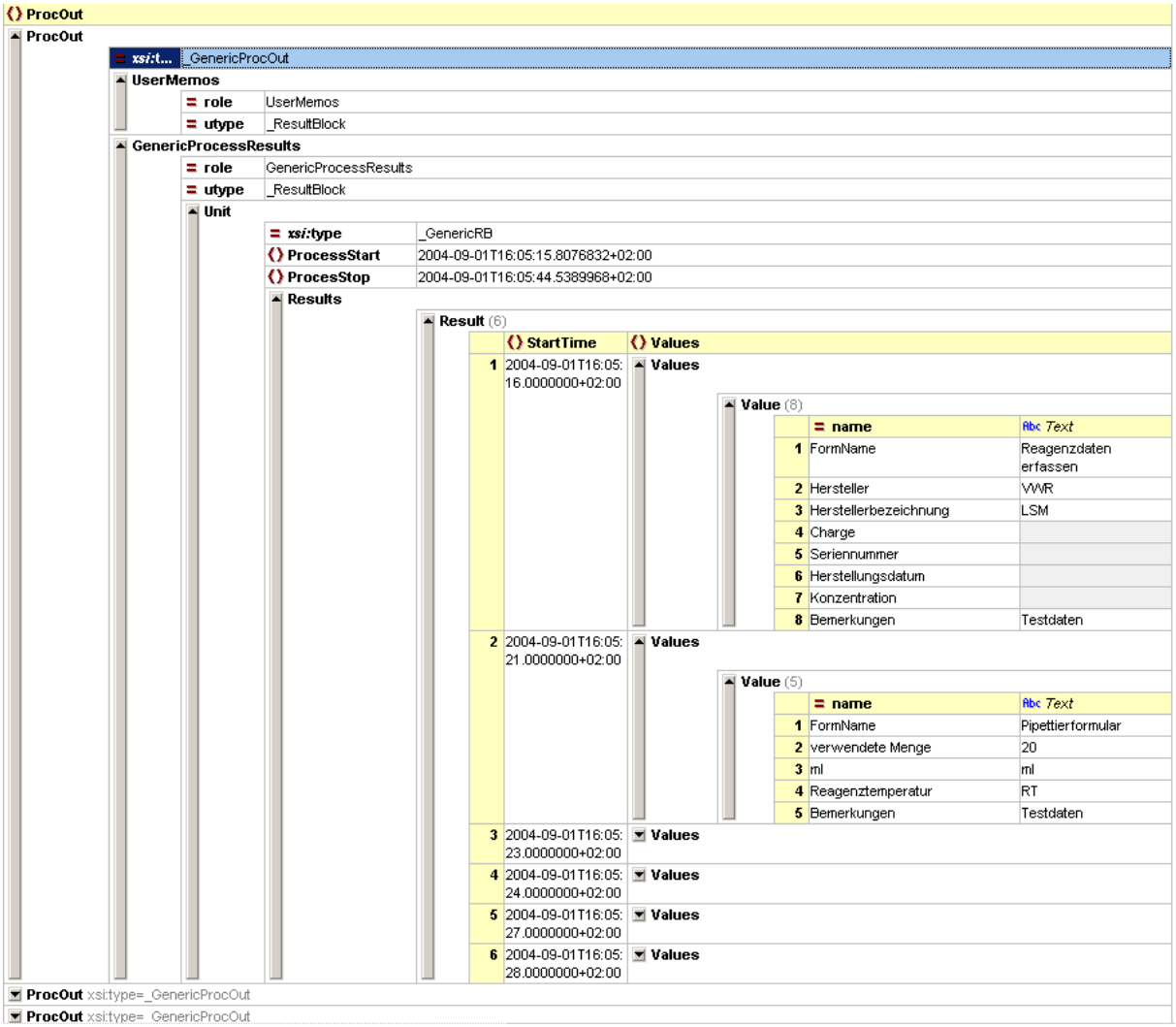

Abbildung 51: auszugsweise Abbildung von Dokumentationsdaten, die durch Verarbeitung von Eingabeformularen durch eine Instanz von ‚Generic Device' akquiriert wurden. Der hier exemplarisch dokumentierte Prozeß besteht aus sechs Aktivitäten. Jeder Aktivität ist ein Ergebnisblock (‚Results') zugeordnet. Dieser Ergebnisblock dokumentiert unter anderem die jeweiligen Eingabefelder des verwendeten Formulars und die vom Laborpersonal eingegebenen Werte unter Verwendung des Elementtyps ‚Value'. Die Elementtypen ‚Results' und ‚Value' sind Bestandteile des Elementtyps ‚UnitType' (siehe Abschnitt 5.2.2.6 und Abbildung 46). Die Felder der Eingabeformulare und testweise eingegebene Daten von zwei Prozeßschritten sind im Detail zu sehen.

Der Elementtyp ‚_GenericProcLog' wurde zusätzlich zu seiner Verwendung in der Datenstruktur für dezentrale Wissensbewahrung dazu verwendet, eine zentral gespeicherte Dokumentation von Einzelprozessen zu realisieren. In Ergänzung zur Akkumulierung der Dokumentation mehrerer Aktivitäten im Elementtyp ‚RunDocumentation' auf dem Chip legt jede Instanz des ‚Generic Device' eine Instanz des ‚_GenericProcLog' für jeweils einen ausgeführten Prozeß zentral in Form einer eigenen XML-Datei ab. Dazu wird ein allen Laborstationen

gemeinsam zugängliches Verzeichnis (bezeichnet als *OnlineDB*) verwendet, dessen Gesamtheit eine vollständige Workflow-Dokumentation darstellt. Die Nomenklatur beim Erzeugen der jeweiligen Dateinamen beinhaltet Informationen zu Datum, Prozeßname und ausführender Station.

Die Integration der in Abschnitt 5.2.2 konzipierten Datenstruktur in das in Kapitel 5.1 beschriebene Testsystem hat die grundsätzliche Machbarkeit des Prinzips *Probe steuert ihre eigene Präparation* unter Verwendung von Datenstrukturen auf dezentralen Speicherchips gezeigt. Außerdem wurde das Proben- und Prozeßwissen, das bei der Abarbeitung von Workflow-Definitionen durch das Testsystem generiert wurde, einheitlich und unmittelbar und unter Einbeziehung des Laborpersonals elektronisch akquiriert und sowohl in der Datenstruktur für dezentrale Wissensbewahrung als auch in einer OnlineDB abgelegt.

Für die Instanziierung der Datenstruktur und für die Integration der Datenstruktur in prototypische Lagertechnologie wurden zusätzliche Applikationen konzipiert, die Thema der folgenden Abschnitte sind.

5.2.4 ‚EurocryoDB' und Instanziierung der Datenstruktur

In Abschnitt 5.2.3 wurde die Interaktion des Testsystems mit der Datenstruktur zur dezentralen Wissensbewahrung prinzipiell beschrieben. Voraussetzungen dafür sind einerseits das Vorhandensein einer Instanz der Datenstruktur auf dem Compact-Flash der Unity und andererseits eine Referenz auf eine Workflow-Definition in dieser Instanz (Elementtyp ‚Run', siehe Abschnitt 5.2.2.5). Diese Voraussetzungen müssen zunächst geschaffen werden, beginnend bei der Auswahl von Workflow-Definitionen.

Die Auswahl einer geeigneten Workflow-Definition für eine Probe ist unter anderem vom Zelltyp abhängig und erfordert somit eine hinreichende Kenntnis wesentlicher Probeneigenschaften, um Fehlpräparationen durch ungeeignete Workflows zu vermeiden. Entsprechendes probenrelevantes Wissen wird dem Laborpersonal der Forschungs- und Demonstrations-Biobank ‚Eurocryo' durch eine Probendatenbank (‚EurocryoDB') zur Verfügung gestellt, so daß die Auswahl einer Workflow-Definition aus ‚EurocryoDB' heraus erfolgen sollte. Dies erfordert:

- Zugriff der ‚EurocryoDB' auf die ‚Evoscreen'-Datenbank
- Auswahl einer geeigneten Workflow-Definition für eine Probe
- Temporäres Speichern der getroffenen Auswahl

Um dies zu realisieren, wurde zunächst die Applikationslogik der EurocryoDB um lesenden Zugriff auf die ‚Evoscreen'-Datenbank erweitert[161]. Eine Auswahlliste der zur Verfügung stehenden Workflow-Definitionen kann aus der Maske für die Bearbeitung der Probendaten heraus aufgerufen werden[162]. Um die Auswahl eines Runs für eine Probe (temporär) in der Eurocryo-DB zu speichern, wurde die Erweiterung der EurocryoDB-Datenbankstruktur um eine zusätzliche Tabelle konzipiert[163]. Die Auswahl eines Runs für eine Probe führt zum Anlegen eines neuen Datensatzes in der Tabelle ‚KRYO_BernsteinSequenz'. Die ID der Probe, für die der Run ausgewählt wurde, wird dort im Feld ‚SampleID' gespeichert, die Probenbezeichnung im Feld ‚SampleName', die ID der Substrate in den entsprechenden Feldern, die ID der Rundefinition im Feld ‚IDSequenz' und der Name des Runs im Feld ‚Bezeichnung'. Die Datensätze der Tabelle ‚KRYO_BernsteinSequenz' werden für die Erzeugung von Runreferenzen im Elementtyp ‚Run' in dezentralen Instanzen der Datenstruktur aus Kapitel 5.2 benötigt, wie im Folgenden beschrieben.

Für das Instanziieren der Datenstruktur auf dem Compact-Flash einer Unity und für das Generieren einer Referenz auf eine Workflow-Definition in der ‚Evoscreen'-Datenbank wurde eine zusätzliche Applikation konzipiert und implementiert, die eine Instanz der Datenstruktur aus Kapitel 5.2 generiert[164] und unter anderem

- in der Tabelle ‚KRYO_BernsteinSequenz' der ‚EurocryoDB' nach Datensätzen für eine bestimmte SampleID sucht und gegebenenfalls die Felder des Datensatzes selektiert
- in der generierten Instanz der Datenstruktur entsprechende Elemente anfügt für
    - SampleID → root/Probe/ProbenID
    - SampleName → root/Probe/ProbenInfo/Beschreibung
    - SubstratID_1[165] → root/Substrat/Substrat_ID
    - IDSequenz → root/Run/Run_ID

---

[161] Voraussetzung ist eine Schnittstelle zwischen ‚EurocryoDB' und ‚Evoscreen'-Datenbank.

[162] Die Probendatenmaske wurde dazu um eine entsprechende Schaltfläche (‚Bernstein-Run') erweitert, die unter Verwendung einer SQL-Abfrage alle Datensätze der Run-Tabelle in einer Auswahlliste ausgibt.

[163] Implementierung: Tabelle ‚KRYO_BernsteinSequenz', u.a. mit den Feldern ‚SampleID', ‚SampleName', ‚SubstratID_1' (bis ‚SubstratID_5'), ‚IDSequenz' und ‚Bezeichnung'. Hier ist anzumerken, daß ‚Sequenz' als Synonym für ‚Run' bzw. ‚Workflow-Definition' zu verstehen ist und nicht im Sinne der Nomenklatur gemäß Kapitel 5.1.

[164] Diese Instanz der Datenstruktur wird mit Hilfe eines geeigneten Schreib-/Lesegerätes auf den Compact-Flash geschrieben.

[165] Der Übersichtlichkeit wegen wird hier nur SubstratID_1 betrachtet. Für jede der maximal fünf SubstratIDs (siehe Abschnitt 5.2.2.3) aus dem Datensatz wird jedoch ein eigenes Element root/Substrat an die Datenstruktur angefügt.

    - Bezeichnung → root/Run/Run_Name
- das Erfassen der Compact Flash-ID erlaubt und in der Instanz der Datenstruktur anfügt:
    - Flash ID → root/CF/CF_ID

Auf diese Weise wurde die in Abschnitt 5.2.3 beschriebene Interaktion zwischen Testsystem und Datenstruktur ermöglicht.

5.2.5 Integration der Datenstruktur in prototypische Lagertechnologie

Grundsätzlich sollen lesende und schreibende Zugriffe in die Datenstruktur für dezentrale Wissensbewahrung auch während der Probenlagerung unter tiefen Temperaturen stattfinden. Konventionelle Lagertechnologie jedoch stellt keine Möglichkeiten für elektronische Konnektierung zur Verfügung. Daher wurde eine prototypische Probenschublade entwickelt, die auf Basis einer integrierten Demultiplexerschaltung das wechselweise Konnektieren mehrerer Compact-Flash-Speichermedien mit einem geeigneten Lese-/Schreibgerät ermöglicht. Diese Probenschublade und das Lesegerät wurden in Kapitel 4.3 beschrieben und in Abbildung 27 illustriert. Um den Inhalt der Datenstrukturen der verschiedenen Unities automatisch anzeigen zu lassen, wurde eine weitere Applikation konzipiert, die

- die Demultiplexerschaltung steuert und jeweils nach Ablauf eines definierten Zeitintervalls die jeweils nächste Position selektiert.
- für die selektierte Position die Datenstruktur einliest und ausgewählte Inhalte darstellt. Die darzustellenden Elemente werden mit Hilfe eines *XML-Stylesheets* selektiert und visuell angeordnet.
- die nacheinander ausgelesenen Positionen der Schublade in einer Übersicht darstellt.
- auf Änderungen in der Belegung der Schublade reagiert, indem sie die Positionen zyklisch abfragt.

Auf Basis dieser Konzeption wurde die Applikation ‚Cryoplex' implementiert, die alle Positionen der Schublade nacheinander ansteuert, die Datenstrukturen der aufgesteckten Compact-Flash-Speicher ausliest und die relevanten Informationen in einem GUI anzeigt. Damit wurde eine erste Integration der Datenstruktur aus Abschnitt 5.2.2 in prototypische Lagertechnologie erreicht.

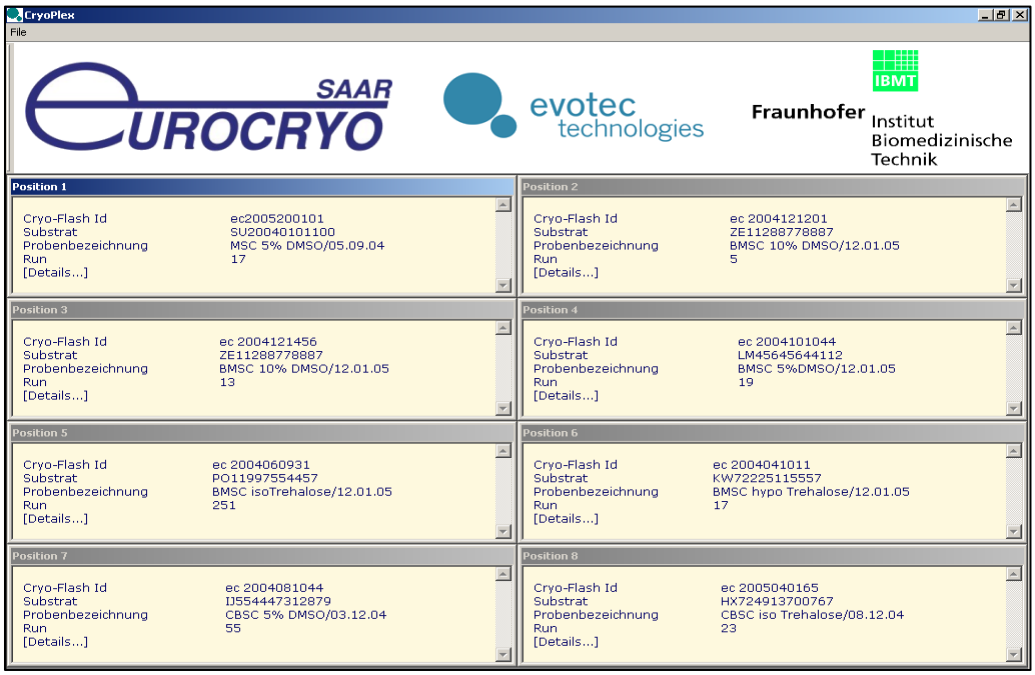

Abbildung 52: Das GUI der Applikation ‚Cryoplex'. Diese Applikation dient der Steuerung der in Abbildung 27 gezeigten Demultiplexer-Elektronik, um selektiv auf die einzelnen Positionen der in Kapitel 4.3 beschriebenen und abgebildeten prototypischen Lagerschublade zugreifen zu können. Wenn die Lagerschublade mit Unities (Substratstapel mit Compact-Flash-Chip) bestückt ist und jeweils eine Instanz der in Abschnitt 5.2.2 konzipierten Datenstruktur auf dem jeweiligen Chip vorhanden ist, wird der Inhalt der Datenstruktur durch die Applikation ‚Cryoplex' ausgelesen und in einem eigenen Bereich des GUI abgebildet. In obiger Abbildung ist das GUI für die Anzeige der Datenstrukturen von acht Substratstapeln konfiguriert. ‚Cryoplex' fragt die einzelnen Positionen der Schublade sequentiell ab und arbeitet iterativ, so dass Änderungen in der Schubladenbelegung automatisch festgestellt und dargestellt werden. Die dargestellten Informationen wie auch das Erscheinungsbild des GUI sind vom verwendeten XML-Stylesheet abhängig. Durch Modifikation des Stylesheets oder durch Verwendung eines anderen Stylesheets kann das Erscheinungsbild verändert werden oder auch andere Informationen innerhalb der Datenstruktur zur Anzeige selektiert werden. Das abgebildete GUI ist prototypisch und wurde im Zusammenhang mit dem Testsystem für Kontrollzwecke und im Zusammenhang mit der prototypischen Lagerschublade zu Demonstrationszwecken eingesetzt.

5.3 Anforderungen an den GHRC-Prototypen

Wie in Kapitel 5.1 beschrieben, bestand die Intention bei Konzeption und Implementierung des Testsystems einerseits in der Evaluierung grundsätzlicher Machbarkeit[166] und Akzeptanz[167] von Benutzerführung[168] durch ein Informationssystem bei der Abarbeitung biomedizinischer Laborprotokolle, andererseits aber auch in der Verwendung des Testsystems als konkrete Grundlage für die Identifizierung weiterer Anforderungen an Folgesysteme. Diese Anforderungen haben sich im Wesentlichen aus funktionalen[169] Limitierungen des Testsystems ergeben, die bei seiner Verwendung deutlich wurden, aber auch anhand neuer Aspekte im Rahmen des in Kapitel 4.2 vorgestellten GHRC-Projektkontextes.

5.3.1 Funktionale Limitierungen des Testsystems

Das in Kapitel 5.1 beschriebene Testsystem wurde in einem Labortrakt des Fraunhofer-IBMT und in den Laboratorien der Forschungs- und Demonstrations-Biobank ‚Eurocryo' der Fraunhofer-Gesellschaft installiert. Beiden Installationen gemeinsam war das in Abschnitt 5.1.2 beschriebene Konzept zur Aufteilung der Laboratorien in verschiedene Laborstationen, demgemäß die jeweilige Geräteausstattung der Stationen für die Gruppierung der einzelnen Aktivitäten der verschiedenen Präparationsprotokolle zu Prozessen und für deren Zuordnung zu Laborstationen[170] maßgeblich ist. Der Labortrakt im Fraunhofer-IBMT wurde in sechs Stationen unterteilt, in den Laboratorien der ‚Eurocryo'-Biobank hingegen wurden drei Laborstationen realisiert, die sich hinsichtlich der zugeordneten Laborgeräte und somit ihrer Funktionalitäten von den Laborstationen im Fraunhofer-IBMT unterschieden.

Die aus der Verwendung der ‚Evoscreen'-Workflow-Struktur resultierenden statischen Gruppierungen von Aktivitäten in Prozesse und die statischen Zuordnungen von Prozessen zu Laborstationen, die im Zusammenhang mit der gezielten Verteilung prozeduralen Wissens innerhalb eines einzelnen Laboratoriums vorteilhaft waren, sind jedoch nur zwischen Laboren

---

[166] Die grundsätzliche Machbarkeit und die Interaktion des Testsystems mit Strukturen zur dezentralen Wissensbewahrung und –akquise wurden in den Kapiteln 5.1 und 5.2 gezeigt.

[167] Siehe hierzu Abschnitt 5.3.2.

[168] Als Ziele von Benutzerführung durch ein WfMS waren neben dem Aspekt papierloser Laborarbeit insbesondere Verbesserungen hinsichtlich Bearbeitungsqualität, Handlungsgüte und Verbesserungen bei Wissensgenerierung und –akquise formuliert worden.

[169] Weitere Limitierungen des Testsystems wurden anhand einer Evaluierung des Systems hinsichtlich ergonomischer Aspekte deutlich. Die Ergebnisse dieser Untersuchungen werden im Zusammenhang mit der Konzeption einer Benutzerschnittstelle in Abschnitt 5.4.6 beschrieben.

[170] Vergleiche dazu die Struktur von ‚Evoscreen'-Workflow-Definitionen in Abschnitt 5.1.2.

mit gleicher Aufteilung und gleicher Anzahl und Ausstattung von Laborstationen kompatibel[171]. Aus den Unterschieden zwischen beiden Installationen des Testsystems resultierte daher die Nicht-Kompatibilität der Workflow-Definitionen der ‚Eurocryo'-Laboratorien mit dem Labortrakt des Fraunhofer-IBMT, und umgekehrt[172]. Um eine Workflow-Definition auf das jeweils andere Labor zu übertragen, mußten sowohl ihre Prozeßdefinitionen wie auch deren Zuordnung zu Laborstationen manuell überarbeitet werden[173]. Ein unmittelbarer Austausch von Workflow-Definitionen zwischen verschiedenen Installationen des Testsystems scheitert somit an der ‚Evoscreen'-Workflow-Struktur, unabhängig von der im Testsystem aus Gründen der Vereinfachung erfolgten lokalen Speicherung von Befehlsparametern[174] und unabhängig von der Verwendung von Referenzen[175] auf Workflow-Definitionen in der ‚Evoscreen'-Datenbank.

Eine weitere gravierende Limitierung des Testsystems resultiert aus der Synchronisierung der Prozesse an den Laborstationen durch den Scheduler ‚Bernstein', der zuvor die Referenzen der einzelnen Prozesse an die zugeordneten Laborstationen[176] verteilt. Da ‚Bernstein' nicht mehrere Runs gleichzeitig synchronisieren kann, kann immer nur eine einzige Workflow-Definition (und damit nur ein einziges Laborprotokoll) pro Zeitpunkt und Labor ausgeführt werden. Alle in einen Run involvierten Laborstationen sind während dieser Zeit blockiert und können nicht anderweitig verwendet werden.

Eine weitere Limitierung existiert hinsichtlich der Akquise von Labordokumentation. Diese erfolgt im Testsystem mit Hilfe vordefinierter Eingabeformulare, die Bestandteil der Applikation ‚Generic Device' sind. Ihr Aufruf und ihre Verarbeitung werden durch Aktivitäten definiert[177]. Die Eingabeformulare wurden anhand repräsentativer Protokolle entworfen, sind aber nicht zwingend für alle künftigen Protokolle geeignet[178]. Für jedes neue Eingabeformular wäre somit eine Erweiterung der Applikation ‚Generic Device' erforderlich.

---

[171] Eine derartige Übereinstimmung verschiedener Laboratorien ist in der Realität nicht gegeben.

[172] Einerseits waren die Zuordnungen der Prozesse zu den Laborstationen nicht übertragbar, und andererseits entsprachen die in den Prozessen definierten Aktivitäten zumindest teilweise nicht den Funktionalitäten der zugeordneten Stationen.

[173] Eine vergleichbare Problematik ergibt sich auch dann, wenn eine Umorganisation der Laborstationen in einem Labor erfolgt oder sich die Geräteausstattung an einer Laborstation ändert. Dann ist das Labor mit seinen eigenen Workflow-Definitionen nicht mehr kompatibel.

[174] Hiermit sind die HTML-Dateien zum Bereitstellen prozeduralen Wissens an Laborpersonal gemeint.

[175] im Elementtyp ‚Run' der Datenstruktur für dezentrale Wissensbewahrung (siehe Kapitel 5.2)

[176] genauer: an die jeweilige Instanz der HumanApplication ‚Generic Device'

[177] vgl. den Befehl ‚ProcessForm' in Kapitel 5.1.

[178] Dies wurde auch im Rahmen der durchgeführten Evaluierung deutlich.

Die rein sequentielle Struktur von ‚Evoscreen'-Workflow-Definitionen, die eine totale Ordnung der einzelnen Aktivitäten eines Präparationsprotokolls darstellt, entsprach den in Kapitel 5.1 betrachteten Laborprotokollen. Sie war für die Vorgaben und die Zielsetzung des Testsystems hinreichend[179]. Im Zusammenhang mit der Aufarbeitung HIV-infizierter Blutproben im Kontext von CAVD und GHRC wurden jedoch komplexere Protokolle relevant. Folgende Abbildung illustriert das vollständige Protokoll zur Aufarbeitung HIV-infizierter Blutproben:

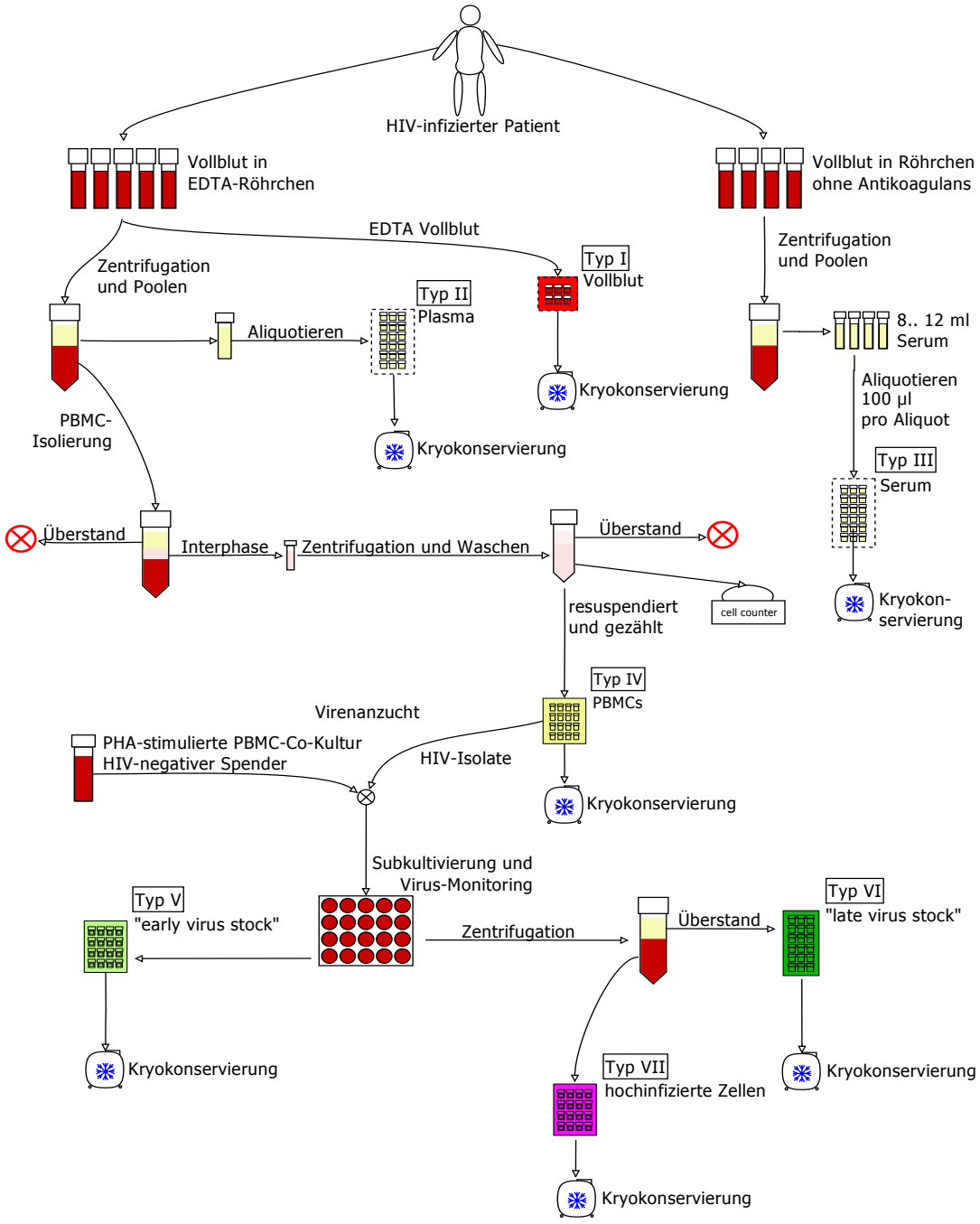

Abbildung 53: Vereinfachte schematische Darstellung der vollständigen Aufarbeitung HIV-infizierter Blutproben mit einer Übersicht der entstehenden Probenprodukte. Alle Probenprodukte sind im Rahmen der Impfstoff-

[179] Siehe dazu auch die Ausführungen in Abschnitt 5.1.2.

forschung bedeutsam. Ihre Langzeitlagerung erfolgt mit Hilfe von Kryokonservierung innerhalb des GHRC-Konsortiums. Bereits bei der Probenentnahme von einem Spender werden verschiedene Aufarbeitungspfade berücksichtigt, die unterschiedliche Blutentnahmeröhrchen (mit bzw. ohne Antikoagulans, hier: EDTA) erfordern. Die unterschiedlichen Präparationspfade führen zu verschiedenen Probenprodukten. Ein Teil der Vollblutprobe in EDTA-Röhrchen kann aliquotiert und kryokonserviert werden (Probenprodukt 1: EDTA-Vollblut). Die restlichen EDTA-Vollblutproben werden unter anderem einer Dichtezentrifugation unterzogen, um Blutplasma zu isolieren und zu kryokonservieren (Probenprodukt 2: Blutplasma). In weiteren Präparationsschritten werden aus dem nach Abnehmen des Blutplasmas übrigen Sediment mononukleäre Zellen (PBMCs) isoliert, die hohe Bedeutung für die Erforschung potentieller Impfstoffe haben (Blutprodukt 4: PBMCs). Auch PBMCs werden mit Hilfe von Kryokonservierung langzeitgelagert. Aus ihnen können unter anderem HIV-Virusisolate gewonnen werden, die durch Subkultivierung mit PBMCs HIV-negativer Spender die Isolation sogenannter *early virus stocks* (Probenprodukt 5: early virus stock) ermöglichen und durch weitere Kultivierung und anschließende Zentrifugation die Isolierung von *late virus stocks* (Probenprodukt 6: late virus stock) und hochinfizierten Zellen (Probenprodukt 7: hochinfizierte Zellen) ermöglichen. Die Vollblutprobe des Spenders in Röhrchen ohne Antikoagulans wird verschiedenen Bearbeitungsschritten unterzogen, um Serum (Probeprodukt 3: Serum) zu isolieren und zu kryokonservieren. Die verschiedenen abgebildeten Präparationspfade sind teilweise unabhängig voneinander, d.h. sie können simultan ausgeführt werden, beispielsweise die Isolation von Serum und das Einfrieren einer EDTA-Vollblutprobe. Beispielsweise können auch die beiden parallelen Pfade zur Isolation von Blutplasma und PBMCs nach einer gemeinsamen Dichtezentrifugation simultan oder nacheinander abgearbeitet werden[180].

Innerhalb des GHRC-Konsortiums soll eine kooperative, dezentrale Aufarbeitung von Blutproben stattfinden, d.h. die verschiedenen Präparationspfade sollen in unterschiedlichen Laboratorien und zu unterschiedlichen Zeitpunkten abgearbeitet werden können. Dies würde ermöglichen, Teile einer Aufarbeitung je nach den Möglichkeiten eines Laboratoriums auszuführen und die verbleibenden Teile in anderen Laboratorien durchzuführen. Voraussetzung dafür sind unter anderem ein Austausch von Proben und Probenprodukten, ein Austausch der Workflowdefinition, des erreichten Abarbeitungsstatus und der bisher akquirierten Dokumentation. Dieser globale Charakter der Aufarbeitung HIV-infizierter Blutproben wird beispielsweise auch anhand der beabsichtigten Interaktion von Primary Sites und GHRC deutlich: vor Ort sollen von den Primary Sites verschiedene Probenprodukte (beispielsweise EDTA-Vollblut, Blutplasma, Serum oder PBMCs) isoliert und zur Lagerung oder weiteren Verarbeitung an das GHRC versendet werden können. Im Gegensatz zu den in Kapitel 5.1 betrachteten rein sequentiellen, lokalen Workflows müssen beim GHRC-Prototypen somit komplexe, globale Workflows berücksichtigt werden und eine Möglichkeit zu ihrem Austausch zwischen verschiedenen Laboratorien und zur verteilten Abarbeitung geschaffen werden. Die Komplexität des in Abbildung 53 schematisch abgebildeten Protokolls wird nachfolgend zusammengefasst:

[180] Abbildung 9 veranschaulicht diese Parallelität der Isolation von Plasma und Blutzellen nach einer gemeinsamen Dichtezentrifugation anhand eines Graphen.

- Es existieren mehrere verschiedene Präparationspfade innerhalb des Protokolls, ausgehend von der Ausgangsprobe oder ausgehend von aus ihr entstandenen Probenprodukten.

- Die unterschiedlichen Präparationspfade dienen der Isolierung oder Herstellung unterschiedlicher Produkte oder wissenschaftlicher Erkenntnisse aus einer Probe oder aus einem Probenprodukt.

- ‚Sample Splitting': Aufteilung einer Probe oder eines Probenprodukts in mehrere Aliquoten, die anschließend unterschiedliche weitere Präparationspfade durchlaufen.

- Bestimmte Aktivitäten (beispielsweise eine Dichtezentrifugation) führen zur Auftrennung einer Probe in verschiedene Probenprodukte, die anschließend unterschiedliche weitere Präparationspfade durchlaufen.

- Die verschiedenen Präparationspfade für eine Probe oder ein Probenprodukt werden nicht ausschließlich nacheinander, sondern teilweise simultan abgearbeitet. Die Aktivitäten einer Präparation sind somit nicht total, sondern partiell geordnet.

- Teilweise sind Aktivitäten abhängig von Ergebnissen vorheriger Aktivitäten.

- Teilweise sind Benutzerentscheidungen für die weitere Bearbeitung erforderlich.

- Berücksichtigen von Bedingungen (beispielsweise Ausführung bestimmter Pfade nur bei entsprechend vorhandenem Probenvolumen; Priorisierung)

- Globaler Charakter: Teile des Protokolls werden räumlich und zeitlich versetzt ausgeführt (kollaborative Bearbeitung in mehreren Laboratorien).

‚Evoscreen' hingegen erlaubt nur rein sequentielle Workflow-Definitionen ohne Kontrollstrukturen und ohne Verzweigungen und ist somit für die Abbildung von Protokollen mit der oben festgestellten Komplexität nicht hinreichend. Der Scheduler ‚Bernstein' synchronisiert ausschließlich die sequentielle Ausführung einzelner Prozesse für eine Probe, unabhängig von den enthaltenen Aktivitäten. Eine inhaltlich so komplexe Protokolldefinition wie oben abgebildet kann in Form von ‚Evoscreen'-Workflow-Definitionen nicht abgebildet, von ‚Bernstein' nicht synchronisiert und von ‚GenericDevice' nicht abgearbeitet werden. Die Anforderungen an den GHRC-Prototypen, die sich aus den beschriebenen Limitierungen des Testsystems ergeben, werden in Abschnitt 5.3.3.1 formuliert.

5.3.2 Limitierungen des Testsystems aus Anwendersicht

Das Testsystem wurde von repräsentativen Mitarbeitern[181] des Fraunhofer-IBMT evaluiert[182], um zusätzlich zu den in Abschnitt 5.3.1 beschriebenen funktionalen Limitierungen auch Limitierungen aus Anwendersicht zu identifizieren, die bei der Konzeption eines Folgesystems relevant sind[183]. Genannt wurden[184]:

- Komplizierte Systembedienung: die Zusammensetzung des Testsystems aus verschiedenen Applikationen (‚Bernstein' für Auswahl und den Start von Runs, ‚Generic Device' für die Abarbeitung der Prozesse) wird als unübersichtlich, umständlich und nicht intuitiv[185] empfunden.

- Fehlen einer Gesamtübersicht[186]: für die Planung der Anwender, das Vorbereiten der einzelnen Präparationsschritte und das Bereitstellen von Materialien und Reagenzien ist aus Sicht der Anwender ein Überblick über das gesamte Protokoll erforderlich. Das Testsystem jedoch informiert nur schrittweise während der Präparation einer konkreten Probe.

- Material- und Reagenzienliste: die Platzierung der Informationen über benötigte Materialien und Reagenzien im jeweiligen Präparationsschritt wird als nicht zweckmäßig beurteilt. Sämtliche Reagenzien und Materialien müssen vor Präparationsbeginn bekannt sein, um spätere Arbeitsunterbrechungen zu vermeiden.

- Unklarheit der Dokumentation: der Benutzer hat keinen Überblick darüber, welche Dokumentationsdaten insgesamt während einer Protokollausführung erfaßt werden.

- Fehlende Orientierung: Anwender möchten jederzeit frühere oder nachfolgende Präparationsschritte einsehen können, um sich zu orientieren.

---

[181] Es handelte sich um 3 promovierte Biologen, 2 Diplom-Biologen und 2 Technische Assistentinnen.

[182] In diesem Zusammenhang wurde das Konzept der Benutzerführung durch ein Informationssystem und der automatisierten Wissensakquise anhand des Testsystems evaluiert und von allen Befragten als geeignet beurteilt, um eine einheitliche und reproduzierbare Protokollausführung und vergleichbare Dokumentation zu erreichen, eine höhere Ausgereiftheit des Systems vorausgesetzt.

[183] Auf die Evaluierung ergonomischer Aspekte geht Abschnitt 5.4.6 ein.

[184] Auffallend war, dass nahezu alle Befragten die gleichen Limitierungen genannt haben.

[185] Mehr zu dieser Thematik im Zusammenhang mit der Evaluierung der Benutzerschnittstelle in Abschnitt 5.4.6.

[186] Eine solche Gesamtübersicht scheitert an der Struktur von ‚Evoscreen'-Workflow-Definitionen. Instanzen des ‚Generic Device' können nur die Aktivitäten der ihnen zugeordneten Prozesse anzeigen, nicht aber eine Gesamtübersicht.

- Risiko irreversibler Fehlbedienung: das irrtümliche Quittieren eines noch nicht abgeschlossenen Präparationsschrittes führt ohne Warnung unmittelbar zum nächsten Präparationsschritt. Eine Rückkehr zum ursprünglichen Schritt ist nicht möglich.

- Zu hoher Zeitaufwand: die unmittelbare Bearbeitung von Eingabeformularen als Voraussetzung für den nächsten Präparationsschritt wird beanstandet. Hinsichtlich zeitkritischer Präparationsschritte müssen Unterbrechungen im Ablauf vermieden werden. Eine flexiblere Möglichkeit der Wissensakquise wird gewünscht.

- Wiederholte Dateneingabe: die Dokumentationsmethodik, beispielsweise für verwendete Reagenzien, wird als ungeeignet empfunden, da das wiederholte Erfassen desselben Reagenz in verschiedenen Protokollschritten zu unnötigen Arbeitsunterbrechungen führt.

5.3.3 Anforderungen an den GHRC-Prototypen

5.3.3.1 Anforderungen aus den identifizierten Limitierungen

Bezug nehmend auf die in Abschnitt 5.3.1 identifizierten Limitierungen des Testsystems lassen sich folgende Anforderungen an den GHRC-Prototypen formulieren:

- In kollaborativen Forschungsszenarien wie beispielsweise GHRC oder CAVD wie auch zwischen den verschiedenen Laboratorien des Fraunhofer-IBMT und der ‚Eurocryo'-Forschungsbank ist eine unmittelbare Übertragbarkeit der Workflow-Definitionen erforderlich. Es hat sich gezeigt, daß eine statische Gruppierung der einzelnen Aktivitäten eines Präparationsprotokolls zu Prozessen, die auf der spezifischen Aufteilung eines Labors und der konkreten Ausstattung von Laborstationen beruht, diese Übertragbarkeit behindert, unabhängig von lokaler oder dezentraler Speicherung der Workflow-Definitionen und eventuell assoziierter Dateien[187]. Gleiches gilt für die statische Zuordnung der Prozesse zu konkreten Laborstationen oder zu konkreten Laborgeräten. Eine wesentliche Anforderung an den GHRC-Prototypen ist somit die Verwendung laborunabhängiger Workflow-Definitionen, um unmitttelbare Übertragbarkeit zu gewährleisten.

- Die Workflow-Definitionen müssen der in Abschnitt 5.3.1 beschriebenen Komplexität genügen, d.h. unter anderem parallele Präparationspfade und Kontrollstrukturen enthalten können. Der GHRC-Prototyp muß mit diesen Workflow-Definitionen kompatibel sein.

---

[187] beispielsweise lokal gespeicherte HTML-Dateien, die im Testsystem zur Interaktion mit den Systembenutzern verwendet wurden.

- Hinsichtlich der durch das *Travelling Chip*[188]-Konzept beschriebenen Aspekte der Wissensbewahrung und Wissensverteilung in kollaborativen Forschungsszenarien ist die vollständige dezentrale Speicherung von Workflow-Definitionen bei der Probe erforderlich.

- Es muß möglich sein, erforderliche Labordokumentation unabhängig von vordefinierten Eingabeformularen zu erfassen, die die Menge der mit dem System kompatiblen Protokolle einschränken.

- Gemäß der Definition in Kapitel 3.1 ist die simultane Ausführung verschiedener Laborprotokolle eines der Merkmale biomedizinischer Forschungslaboratorien. Die Beschränkung auf einen einzigen Workflow und die Blockierung der Laborstationen durch das Testsystem widerspricht dieser Definition und den Erfordernissen. Der GHRC-Prototyp darf diese Limitierungen nicht aufweisen. Mehrere unterschiedliche Protokolle müssen zeitgleich im selben Labor abgearbeitet werden können.

- Kollaborative Forschungsszenarien erfordern Unterbrechungen von Präparationen und ihre spätere Wiederaufnahme. Der GHRC-Prototyp muß die Wiederaufnahme von Workflow-Instanzen exakt an der Stelle erlauben, wo sie zuvor unterbrochen wurden.

5.3.3.2 Anforderungen aus Anwendersicht

In diesem Abschnitt werden die von potentiellen Anwendern in Befragungen definierten Anforderungen an den GHRC-Prototypen aufgelistet, zusätzlich zu den von ihnen in Abschnitt 5.3.2 identifizierten Limitierungen des Testsystems. Ihre Berücksichtigung ist für die Akzeptanz des Systems essentiell:

- Das System soll nicht dirigieren. Insbesondere wollen die Anwender selbst bestimmen, an welchem Arbeitsplatz sie die jeweiligen Protokollschritte bearbeiten werden.

- Abweichungen von den Instruktionen der einzelnen Aktivitäten müssen dokumentiert werden können.

- Es muß aus Effizienzgründen möglich sein, die Proben mehrerer Spender in derselben Workflow-Aktivität einer Workflow-Instanz zu bearbeiten. Voraussetzung ist, dass es sich um den gleichen Probentyp im gleichen Bearbeitungsstadium handelt, beispielsweise um EDTA-Vollblut oder Blutplasma.

---

[188] Das *Travelling Chip*-Konzept [Durst 2004] bezeichnet die Idee, daß jede Probe bei Probenaustausch und Transport von einem Speichermedium ‚begleitet' wird, das der Wissensbewahrung, der Wissensverteilung und der Wissensakquise dient.

- Proben müssen identifiziert werden können. Der Inhalt eines Röhrchens muß nachvollziehbar sein.
- Das System soll helfen, Fehlpräparationen zu vermeiden und ein Vergessen einzelner Proben zu verhindern.

5.3.3.3 Anforderungen aus Regeln der Laborarbeit und projektbezogene Vorgaben

Es wurden folgende weitere Anforderungen definiert:

- Die Anzahl der Proben wird überschaubar sein, so daß auf eine High-Throughput-Funktionalität verzichtet werden kann (weniger als 100 Proben / Tag).
- Jeder Anwender arbeitet nur ein Protokoll pro Zeitpunkt ab.
- Jeder Anwender bearbeitet pro Zeitpunkt nur Proben eines Typs im gleichen Bearbeitungsstadium, beispielsweise zentrifugiertes EDTA-Vollblut. Es kann sich aus Effizienzgründen um Proben verschiedener Spender handeln.
- Pro Zeitpunkt arbeitet maximal eine Person pro Laborstation.
- An jeder Station wird pro Zeitpunkt nur ein einziges Protokoll abgearbeitet.
- An jeder Station befinden sich nur Proben, die auf die gleiche Weise präpariert werden.
- Die Proben werden von Personen transportiert.
- Eine Präparation setzt sich in der Regel aus halbautomatisierten und manuellen Schritten zusammen.
- Die meisten Geräte werden vollständig von Personen bedient.
- Es sollen die in Kapitel 4.3 beschriebenen diskreten Probenröhrchen mit integriertem seriellem Flash-Speicherchip und zusätzlich Standard-Laborröhrchen verwendet werden.
- Jeder Präparations-Fortschritt soll an die ‚Eurocryo-DB' gemeldet werden.

5.3.3.4 ‚Implizite' Anforderungen

Bei der Befragung der potentiellen Anwender zu den Anforderungen an den GHRC-Prototypen wurde deutlich, daß die Verbalisierung ihres Expertenwissens hinsichtlich der Ausführung von Laborprotokollen und der daraus resultierenden Anforderungen viele Details

unberücksichtigt läßt und große Mühe bereitet. Da die befragten Biologen aber regelmäßig Laborprotokolle erfolgreich und vollständig abarbeiten, verfügen sie offensichtlich unbewußt über das erforderliche Wissen, das durch Routinisierung ehemals bewußter, expliziter Prozesse entstanden sein muß[189]. Dies deckt sich mit der Definition impliziten[190] Wissens und seiner schwierigen Explizierbarkeit. Der GHRC-Prototyp muß auch denjenigen Anforderungen genügen, die im Zusammenhang mit der Nutzung impliziten Wissens während einer Präparation stehen, die aber in den Befragungen nicht formuliert werden konnten. Um auch diese Anforderungen an den GHRC-Prototypen zu identifizieren, wurde die Protokollausführung einer PBMC-Isolierung mit Hilfe einer Videokamera begleitet. Das entstandene Videomaterial wurde anschließend auf ‚implizite' Anforderungen hin untersucht. Folgende Tabelle korreliert Protokollinstruktionen mit Beobachtungen und zeigt die daraus abgeleiteten Anforderungen an den GHRC-Prototypen:

Anweisung	**Tatsächlicher Arbeitsschritt**	**Identifizierte Anforderungen**
Falcons mit Ficoll vorbereiten	[Sterilwerkbank] Es werden acht Falcon-Zentrifugenröhrchen bereitgestellt. Sie werden direkt aus der Flasche mit jeweils etwa 20 ml Lymphozyten-Separationsmedium ‚Ficoll' befüllt.	Die Reagenzdaten des Lymphozyten-Separationsmediums müssen erfaßt werden. Werden mehrere Flaschen des Mediums benötigt, müssen die Daten mehrerer Flaschen erfaßt werden können. Dabei sind die Ausführungen in Abschnitt 5.3.2 hinsichtlich der Erfassung von Reagenzien zu berücksichtigen.
Vollblut / Buffycoat im Verhältnis 1:1 mit PBS (bei Raumtemperatur) in 50ml-Falcon verdünnen, dann mischen: 2-3x mit Stabpipette vorsichtig hoch und runter ziehen	[Sterilwerkbank] Blutbeutel aufschneiden. Blut wird in eine Zellkulturflasche geschüttet. Es sind knapp 100ml geschätztes Volumen. In die Zellkulturflasche wird das gleiche Volumen an PBS hinzugeschüttet. Die Flasche wird verschlossen und von Hand bewegt, nicht geschüttelt.	Die Reagenzdaten des PBS müssen dokumentiert werden. Auch hier muß es möglich sein, die Daten mehrerer benötigter Flaschen zu dokumentieren. Die IDs der zu präparierenden Blutproben müssen mit Hilfe des Systems dokumentiert werden können. Die ID der jeweiligen Blutprobe muß auf die Zellkulturflasche übertragen werden, damit ihr Inhalt nachvollziehbar ist. Während im Film nur eine einzige Blutprobe verwendet wird, werden oftmals viele Blutproben verschiedener Spender simultan präpariert. Je nach Volumen der einzelnen Ausgangsproben variiert das verwendete Gefäß. In der Regel verwendet man pro Spender ein Zentrifugenröhrchen. Alle diese

[189] Siehe hierzu auch [Myers 1992].

[190] Vergleiche hierzu auch Kapitel 2.1.

		Röhrchen müssen mit der ID der jeweiligen Ausgangsprobe beschriftet werden.
20ml Ficoll / LSM Lymphocyte Separation Medium (PAA) mit 30ml Blut/PBS Gemisch vorsichtig und langsam überschichten	[Sterilwerkbank] Die (oben) mit jeweils 20ml Ficoll vorbereiteten Falcons werden bereitgestellt und alle gleichzeitig geöffnet. Eine Pipette wird vorbereitet, eine Automatikpipette auf langsamste Geschwindigkeit eingestellt, um Vermischen während des Überschichtens zu vermeiden. Phasentrennung zwischen Separationsmedium und Blut ist erforderlich für die folgende Dichtezentrifugation. Das Blut/PBS-Gemisch wird geöffnet. Die Pipette wird mit 30 ml befüllt. Das Falcon-Röhrchen wird sehr schräg gehalten und das Blut/PBS wird langsam auf die innere Wand des Röhrchens abgegeben. Es legt sich als dünner Film über das PBS. Es werden die ganzen 30 ml Gemisch aus der Pipette in das Röhrchen abgegeben. Mit derselben Pipette wird der Vorgang für die anderen Röhrchen wiederholt. Die Zellkulturflasche wird während der ganzen Zeit bewegt, um ein Absetzen der Zellen zu verhindern. Beobachtung: Nach Überschichten von 4 vollständig befüllten Röhrchen reicht am Schluß das verbliebene Blut/PBS-Gemisch nicht mehr aus, um die Volumina der vorgelegten Röhrchen auf 50 ml aufzufüllen. Deswegen wird das restliche Blut/PBS-Gemisch erneut mit PBS verdünnt, in diesem Fall zusätzlichen 5 ml PBS. Es werden nun noch zwei Röhrchen aufgefüllt, allerdings haben diese weniger Volumen als die andern vier. Dies ist allerdings irrelevant, da man nur an einem bestimmten Zelltyp (PBMC) interessiert ist, der nach der späteren	Es ist erforderlich, die ID des jeweiligen Blut/PBS-Gemisches auf das vorbereitete und mit Ficoll befüllte Zentrifugenröhrchen zu übertragen. Es muß möglich sein, mehrere Zentrifugenröhrchen mit dem Blut/PBS-Gemisch eines Spenders zu überschichten, wenn entsprechend viel Probenvolumen vorhanden ist. Daher muß das System das Übertragen der ID des Blut/PBS-Gemisches auf entsprechend viele mit Ficoll vorbereitete Röhrchen erlauben. Die Anzahl der so entstehenden Aliquoten ist nicht von vornherein kalkulierbar, muß aber mit Hilfe des Systems dokumentiert werden können. Werden die Monozyten mehrerer Spender isoliert, müssen die überschichteten Röhrchen anhand der verschiedenen Spender unterschieden werden können. Für jede ProbenID muß letztlich die Anzahl der mit ihrem Blut/PBS-Gemisch erfolgreich überschichteten Röhrchen dokumentiert werden (siehe auch Abbruchkriterium). Sollte ein Blut/PBS-Gemisch unbrauchbar werden, so muß dieser Verlust im System dokumentiert werden können unter Verwendung eines Zeitstempels. Die erfolgreiche Abarbeitung eines Blut/PBS-Gemischs muß im System ebenfalls mit Zeitstempel dokumentiert werden können.

	Zentrifugation abgesaugt und mit den anderen PBMCs der anderen Falcons mit gleicher Proben-ID gepoolt (zusammengeschüttet) wird. Insgesamt wurden nun sechs der ursprünglich acht vorgelegten Röhrchen überschichtet. Die Röhrchen werden nun verschlossen. In den Röhrchen, die als erste befüllt wurden, setzen sich aufgrund der Wartezeit un der Gravitationskraft bereits erste Erythrozyten nach unten ins Separationsmedium ab. Abbruchkriterium: Wenn sich während des Überschichtens Blut und vorgelegtes Ficoll miteinander vermischen, kann man dieses Röhrchen wegwerfen.	
Zentrifugieren 30min, 2000rpm (~ 940 g), RT, ohne Bremse	[Sterilwerkbank] Transport der überschichteten Zentrifugenröhrchen zu einer Zentrifuge [Zentrifuge] Beladen der Zentrifuge; der Rotor verfügt insgesamt über 4 Halter für jeweils 4 Falcons. Die sechs Röhrchen werden auf zwei Halter zu jeweils 3 Röhrchen verteilt. Die Zentrifuge wird manuell parametriert und gestartet. Eine Automatisierung ist nicht möglich, da nicht erlaubt („elektrisch betriebene Schleudermaschine“) Die Zentrifugation ist nach etwa 45 Minuten beendet (inkl. Auslaufphase). Die zentrifugierten Falcons werden aus der Zentrifuge entnommen und die Zentrifuge sofort auf 4°C programmiert zur späteren Benutzung.	Das System muß den Anwender über die anstehende Zentrifugation informieren. Der erreichte Zustand der Präparation und die Anzahl der überschichteten Zentrifugenröhrchen pro ProbenID (Aliquoten) müssen dem System an der Zentrifuge zur Verfügung stehen. Die erfolgreiche Zentrifugation muß für jedes zentrifugierte Röhrchen mit einem Zeitstempel dokumentiert werden können. Bei der Zentrifugation mißlungene Probenröhrchen müssen mit einem Zeitstempel versehen dokumentiert und verworfen werden können. Eine Dokumentation der Zentrifugenparameter ist erforderlich. Diese kann entweder automatisch über eine Schnittstelle oder manuell erfaßt werden. Eine Vorschau innerhalb des Protokolls ist sinnvoll, um einen späteren Zentrifugationsschritt bereits an dieser Stelle vorbereiten zu können.
Interphase aller Röhrchen vorsichtig abnehmen und in	[Sterilwerkbank] Nach der Dichtezentrifugation befinden sich im Röhrchen 4 Phasen	Das System muß den Anwender über die anstehende Arbeit an einer Sterilwerkbank informieren.

50ml-Falkons überführen	übereinander (von oben nach unten aufgezählt): - gelbliche Phase: Plasma und Thrombozyten - schmale weißliche Bande: mononukleäre Zellen (Interphase) - farblose Bande: Ficoll - sedimentierte Erythrozyten und Granulozyten Es werden nun die Interphasen der sechs Röhrchen entnommen und auf 4 neue Falcons verteilt, weil die Interphase nach Sichtprüfung sehr viele PBMCs zu enthalten scheint. Um akzeptable PBMC-Konzentrationen zu erreichen, benutzt man hier von vornherein mehrere Röhrchen.	Der erreichte Zustand der Präparation und die Anzahl der zentrifugierten Aliquoten pro ProbenID müssen dem System an der Sterilwerkbank zur Verfügung stehen. Das System muß *Aliquot-Pooling* unterstützen, d.h. das Zusammenführen mehrerer Aliquoten desselben Typs mit derselben ProbenID (= vom selben Spender): Übertragen der ProbenID eines Aliquots auf ein (oder mehrere) neue(s) Röhrchen und anschließendes Umfüllen der relevanten Phase (hier: der Interphase) jedes Aliquots mit derselben ProbenID in diese(s) Röhrchen. Die Anzahl der tatsächlich pro ProbenID benutzten Zielröhrchen ist nicht vorab definiert und kann vom Benutzer bestimmt werden. Die erfolgreiche Überführung jedes Aliquots muß individuell mit einem Zeitstempel dokumentiert werden können. Eine mißlungene Überführung eines Aliquots muß individuell mit einem Zeitstempel dokumentiert werden können. Das Aliquot Pooling muß vom System für die Aliquoten der unterschiedlichen ProbenIDs unterstützt werden. Die Anzahl der entstandenen Zielröhrchen pro ProbenID (‚Interphase-Aliquoten') muß mit Hilfe des Systems dokumentiert werden können.
Auffüllen mit PBS (4°C) auf 50ml	Die 4 Interphase-Aliquoten werden nun mit 4°C kaltem PBS auf jeweilige Volumina von 50 ml aufgefüllt.	Die Reagenzdaten des PBS müssen dokumentiert werden können. Auch hier muß es wieder möglich sein, die Daten mehrerer benötigter Flaschen zu dokumentieren.
Zentrifugieren: 5min, 1200rpm, 4°C, Bremse ein	[Sterilwerkbank] Transport von 2 der 4 aufgefüllten Probenröhrchen zu einer Zentrifuge. Wegen der großen Zellzahl werden lediglich zwei der vier Röhrchen verwendet. [Zentrifuge]	Das System muß den Anwender über die anstehende Zentrifugation informieren. Es muß möglich sein, Röhrchen aus einem Präparationsschritt zu entnehmen. Diese Abweichung muß dokumentiert werden können. Der erreichte Zustand der Präparation

	ist auf 4°C vorgekühlt (siehe oben) Die Zentrifuge wird manuell parametriert und gestartet.	und die Anzahl der zu zentrifugierenden Zentrifugenröhrchen pro ProbenID (Aliquoten) müssen dem System an der Zentrifuge zur Verfügung stehen. Die erfolgreiche Zentrifugation muß für jedes zentrifugierte Röhrchen mit einem Zeitstempel dokumentiert werden können. Bei der Zentrifugation mißlungene Probenröhrchen müssen mit einem Zeitstempel versehen dokumentiert und verworfen werden können. Eine Dokumentation der Zentrifugenparameter ist erforderlich (siehe oben).
Überstand vorsichtig abnehmen und Pellet aufklopfen	[Sterilwerkbank] Eine Absaugvorrichtung mit Pipette wird zum Abnehmen des Überstands verwendet. Das Pellet (durch Zentrifugation verklumpte Zellen am Boden des Röhrchens) wird von Hand aufgeklopft (in sich gelockert, damit die Zellen frei werden) durch Schütteln des Röhrchens. Beide Pellets werden in ein neues Zentrifugenröhrchen zusammengeschüttet.	Das System muß den Anwender über die anstehende Arbeit an einer Sterilwerkbank informieren. Der erreichte Zustand der Präparation und die Anzahl der zentrifugierten Aliquoten pro ProbenID müssen dem System an der Sterilwerkbank zur Verfügung stehen. Das System muß das Zusammenführen der Pellets jeweils einer ProbenID in ein oder mehrere neue Röhrchen unterstützen. Auch dieses Pooling erfordert das Übertragen der ProbenID auf ein oder mehrere neue Röhrchen und das anschließende Umfüllen der Pellets jedes Aliquots mit derselben ProbenID in diese(s) Röhrchen. Die Anzahl der tatsächlich pro ProbenID benutzten Zielröhrchen ist nicht vorab definiert und kann vom Benutzer bestimmt werden. Die erfolgreiche Überführung des Pellets jedes Aliquots muß mit einem Zeitstempel dokumentiert werden können. Eine mißlungene Überführung eines Pellets muß individuell mit einem Zeitstempel dokumentiert werden können. Das Pooling muß vom System für die Aliquot-Pellets der unterschiedlichen

		ProbenIDs unterstützt werden. Die Anzahl der entstandenen Zielröhrchen pro ProbenID (‚Pellet-Aliquoten') muß mit Hilfe des Systems dokumentiert werden können.

Tabelle 9: Identifizierung ‚impliziter' Anforderungen an den GHRC-Prototypen. Bei der Benutzerbefragung hinsichtlich der Anforderungen an das System aus Anwendersicht war deutlich geworden, daß die Verbalisierung prozeduralen Expertenwissens Schwierigkeiten bereitet und lückenhaft ist. Die befragten Personen verfügen aber offensichtlich über das erforderliche Wissen, können es jedoch nur unzureichend ausdrücken. Dies deckt sich mit der Definition impliziten Wissens und seiner schwierigen Explizierbarkeit. Die Tabelle stellt die Anweisungen des Protokolls gegenüber mit Beobachtungen bei der Protokollabarbeitung, die mit der Videokamera aufgezeichnet wurden und anhand tatsächlicher Arbeitsschritte implizites Handlungswissen aufzeigen. Auf Basis dieser Beobachtungen konnten weitere Anforderungen (‚implizite Anforderungen') an den GHRC-Prototypen identifiziert werden.

5.4 Konzeption des GHRC-Prototyps

Im Folgenden wird die Konzeption des GHRC-Prototyps beschrieben, die auf Basis der in Abschnitt 5.3.3 identifizierten Anforderungen erfolgt ist.

5.4.1 Aufbau aus autonomen Laborstationen

In Abschnitt 5.3.1 wurde deutlich, daß (a) eine statische Unterteilung von Laborprotokollen in mehrere Prozesse, die anhand der Funktionalitäten von bestimmten Laborstationen erfolgt war, und (b) die statische Zuordnung dieser Prozesse zu den jeweiligen Laborstationen die unmittelbare Austauschbarkeit von Protokollen zwischen verschiedenen Laboratorien stark einschränken. Eine solche Repräsentation gemäß der ‚Evoscreen'-Workflow-Struktur war ursprünglich für die Synchronisierung verschiedener automatisierter Laborgeräte in High-Throughput-Screening-Modulen durch den Scheduler ‚Bernstein' konzipiert worden und konnte im Testsystem dazu verwendet werden, prozedurales Wissen gezielt an Laborstationen zu präsentieren. In biomedizinischen Forschungslaboratorien (gemäß der Definition in Kapitel 3.1) sollen jedoch intelligente Anwender, die die einzelnen Stationen und ihre Ausstattung kennen, durch ein Workflow-Management-System unterstützt werden. Eine der Anforderungen aus Anwendersicht ist, selbst entscheiden zu wollen, an welcher Laborstation der jeweilige Protokollschritt ausgeführt wird. Eine Dirigierung[191] der Anwender zu Laborstationen, wie sie im Testsystem durch den Scheduler ‚Bernstein' erfolgt war, wird von den Anwendern strikt abgelehnt. Da gemäß der projektbezogenen Vorgaben in Abschnitt 5.3.3.3 auch keine

[191] Die bemängelte Dirigierung ist das Ergebnis der Gruppierung von Aktivitäten in Prozesse und deren statische Zuordnung zu bestimmten Laborstationen, analog zur statischen Zuordnung von Geräteprozessen zu Geräten eines HTS-Moduls.

High-Throughput-Funktionalität benötigt wird, kann auf ‚Bernstein' gänzlich verzichtet werden. Dies kommt der Forderung nach Reduzierung der Systemkomplexität entgegen, die von Anwendern formuliert worden war. Auf eine Gruppierung von Aktivitäten zu Prozessen und auf eine Zuordnung zu Laborstationen kann daher in den Workflow-Definitionen künftig verzichtet werden. Dadurch werden die in Abschnitt 5.3.1 beschriebenen, aus der ‚Evoscreen'-Workflow-Struktur resultierenden Ursachen für eine Nicht-Übertragbarkeit von Workflow-Definitionen zwischen verschiedenen Laboratorien behoben. Der GHRC-Prototyp kann somit als ein System aus mehreren Laborstationen (mit jeweils einer Instanz einer Stationensoftware) konzipiert werden, die ohne übergeordneten Scheduler betrieben werden. Insbesondere ist dadurch auch die geschilderte Problematik der Blockierung aller Stationen durch Abarbeitung einer einzigen Workflow-Definition behoben, die sich aus der Verwendung von ‚Bernstein' ergeben hatte. Dennoch muß das Verhalten der einzelnen Laborstationen durch die jeweilige Workflow-Definition gesteuert werden. Diesem Aspekt widmet sich der folgende Abschnitt.

5.4.2 Workflow-Definitionen und Workflow-Engines

Die Inkompatibilität des Testsystems mit komplexen Laborprotokollen war einerseits Resultat der fehlenden Möglichkeit, diese Protokolle in geeignete Workflow-Definitionen in ‚Evoscreen'-Struktur zu überführen, andererseits aber auch Resultat der fehlenden Fähigkeit des Systems, entsprechend komplexe Workflow-Definitionen abzuarbeiten.

Die Abarbeitung von Workflow-Definitionen erforderte beim Testsystem zwei verschiedene Applikationen: (1) einen übergeordneten System Controller (der unabhängig vom Inhalt der Prozesse agierte, siehe Kapitel 5.1) für die Verteilung der Prozesse an die Stationen und für die Synchronisierung ihrer Abarbeitung und (2) Instanzen der HumanApplication ‚GenericDevice' zur sequentiellen Abarbeitung der einzelnen Aktivitäten von Prozessen. Die definierbaren Aktivitäten waren somit nur durch den Befehlssatz von ‚Generic Device' beschränkt. Dieser umfaßt nur wenige Befehle, beispielweise zur Präsentation prozeduralen Wissens an die Anwender unter Verwendung assoziierter Dateien oder zum Verarbeiten vordefinierter Eingabeformulare. Befehle zur Definition von Kontrollstrukturen (beispielsweise if...then...else oder while-Schleifen) oder Funktionen zur Definition paralleler Präparationspfade sind in ‚Generic Device' nicht implementiert[192] und stehen daher auch für die Definition von Aktivitäten nicht zur Verfügung. Um komplexe Laborprotokolle in Form einer

[192] Bei der Konzeption des Testsystems wurde die Funktionalität der HumanApplication der rein sequentiellen Definitionsstruktur der in Abschnitt 5.1.1 analysierten Laborprotokolle angepaßt. Es wurden nur die dafür erforderlichen Funktionen implementiert.

Workflow-Definition repräsentieren zu können, ist eine entsprechend ausdrucksstarke *Workflow-Beschreibungssprache* erforderlich (siehe auch die Abschnitte 2.2.4 und 5.5.1).

Deutlich über den Befehlssatz von ‚GenericDevice' und der rein sequentiellen Abarbeitung von Aktivitäten hinausgehend, benötigt die Stationensoftware des GHRC-Prototyps dann die entsprechende Funktionalität zum Abarbeiten solch komplexer Workflow-Definitionen und der darin enthaltenen Aktivitäten. Eine solche Funktionalität ist in den zu Workflow-Beschreibungssprachen gehörenden *Workflow-Engines* implementiert. Da die Stationensoftware ohne übergeordneten System Controller betrieben werden soll (siehe auch Abschnitt 5.4.1), muß sie einen lokalen System Controller in Form einer geeigneten[193] Workflow-Engine enthalten, die das Verhalten der jeweiligen Instanz der Stationensoftware durch lokale Abarbeitung der Workflow-Definition und ihrer Aktivitäten bestimmt. ‚Geeignet' bedeutet im Zusammenhang mit der Stationensoftware des GHRC-Prototyps, dass:

- die Workflow-Engine in die Stationensoftware integrierbar ist.
- die zugehörige Workflow-Beschreibungssprache vorgefertigte, instanziierbare und parametrierbare Bausteine für Aktivitäten enthält (beispielsweise in Form einer Activity library), die für das Erstellen von Workflow-Definitionen genutzt werden können, um komplexe Laborprotokolle elektronisch zu repräsentieren, und daß die Workflow-Beschreibungssprache um benötigte Bausteine erweiterbar ist.
- die Workflow-Engine alle benötigten Aktivitäten (auch die selbst entworfenen) abarbeiten kann.
- daß die Workflow-Definitionen vollständig dezentral gespeichert werden können, ohne assoziierte Dateien zu benötigen oder auf Datenbankinhalte zu referenzieren.[194]
- die Workflow-Beschreibungssprache es ermöglicht, die zu akquirierenden Daten unabhängig von vordefinierten Formularen unmittelbar in der Workflow-Definition festzulegen.[195]
- der erreichte Status der Abarbeitung einer Workflow-Instanz bei der Probe gespeichert werden kann[196]

---

[193] Die Eignung konkreter Beschreibungssprachen wird in Abschnitt 5.5.1 evaluiert.

[194] Dieser Aspekt ist hinsichtlich der Realisierung des Travelling-Chip-Konzepts und im Zusammenhang mit der Langzeitlagerung von Proben von hoher Bedeutung. Er entspricht dem in Kapitel 4.5 formulierten Prinzip der Kopplung von Probe und (vollständigem) Protokoll. Dadurch wird es möglich, neue Protokolle in Laboratorien durch einfachen Probenaustausch zu etablieren.

[195] Siehe auch die Limitierung des Testsystems in Abschnitt 5.3.1 durch vordefinierte Eingabeformulare.

- die unter Verwendung der Workflow-Beschreibungssprache definierten Workflow-Definitionen den in Abschnitt 5.3.3 formulierten Anforderungen genügen.
- die Workflow-Beschreibungssprache langzeit-kompatible[197] Workflow-Definitionen ermöglicht.

Die Integration einer geeigneten Workflow-Engine direkt in die Stationensoftware ermöglicht somit die Steuerung des Verhaltens der einzelnen Instanzen der Stationensoftware durch lokale Abarbeitung von komplexen Workflow-Definitionen.

5.4.3 Systeminteraktion mit dezentralen Speicherchips

Dieser Abschnitt beschreibt die erforderliche Interaktion der autonomen Laborstationen mit dezentralen Speicherchips, mit dezentralen Workflow-Definitionen und mit dezentraler Dokumentation.

Die Integration dezentraler Speichermedien beschränkt sich gegenwärtig auf die folgenden Labor-relevanten *Cryo-Devices*:

(a) diskrete Substrate[198] für die Kryokonservierung von Proben,

(b) konventionelle Kryoröhrchen[199] unter Verwendung eines Adapters mit Chip[200],

(c) Lagerboxen mit gemeinsamen Speicherchips für mehrere Probenröhrchen mit integriertem oder adaptiertem Speicherchip (siehe auch Kapitel 4.4).

Proben befinden sich nur unmittelbar vor, während oder unmittelbar nach einer Kryokonservierung in Probengefäßen mit integriertem Speicherchip oder mit ‚Portlink'. Während der Laborarbeit befinden sich Proben gegenwärtig in Standard-Probengefäßen (beispielsweise Zentrifugenröhrchen oder Zellkulturflaschen). Dies ist jedoch hinsichtlich des Zwecks der

---

[196] Dies ist erforderlich (1) hinsichtlich kollaborativer Probenbearbeitung und damit verbundener notwendiger Unterbrechbarkeit und späteter Wiederaufnahme der Präparation und (2) wegen des Konzepts autonomer Laborstationen, zwischen denen der erreichte Status ausgetauscht werden muß. Der erreichte Präparationsstatus muß an der aktuellen Laborstation bei der Probe gespeichert werden und an der nächsten Station zur Verfügung stehen, um die Workflow-Instanz an der richtigen Stelle wiederaufnehmen zu können.

[197] Siehe hierzu auch die Anmerkungen zur Langzeit-Kompatibilität von XML-Dokumenten.

[198] ‚Icebreaker', siehe Kapitel 4.3.

[199] Ob weitere Probengefäße, die nur während der Präparation benötigt werden und daher Verbrauchsmaterialien mit nur kurzer Nutzungsdauer sind, künftig mit Speicherchips ausgestattet werden, hängt von technologischen Anforderungen wie auch von wirtschaftlichen Überlegungen ab.

[200] ‚Portlink', siehe Kapitel 4.3.

Speicherchips als Werkzeug zur dezentralen Wissensbewahrung über lange Zeiträume hinreichend.

Die in Kapitel 4.5 formulierten Prinzipien sind auch unter Verwendung von Standard-Laborgefäßen realisierbar. In diesem Zusammenhang sind die projektbezogenen Vorgaben (siehe Abschnitt 5.3.3.3) von Bedeutung, gemäß denen (1) alle Proben, die sich an derselben Station befinden, dem gleichen Protokollschritt zu unterziehen sind, und (2) sich an einer Laborstation pro Zeitpunkt nur Proben des gleichen Typs und gleichen Bearbeitungszustandes (beispielsweise zentrifugierte EDTA-Vollblutproben) befinden. Daraus folgt einerseits, daß man für alle diese Proben einen gemeinsamen dezentralen Speicherchip nutzen kann, der unter anderem die für diese Proben gültige Workflow-Definition enthält. Daraus folgt andererseits, daß man für den GHRC-Prototypen die in biomedizinischen Laboratorien verbreitete plattenbasierte Arbeitsweise zu Grunde legen kann. Diese besteht im Wesentlichen in der Organisation mehrerer Proben gleichen Typs und gleichen Bearbeitungsfortschritts, die gleichartig zu bearbeiten sind, auf einer gemeinsamen Trägerplatte[201]. Plattenbasierte Probenorganisation und gemeinsamer Speicherchip können in der Konzeption von Trägerplatten mit gemeinsamem Speicherchip[202] für diskrete Standard-Laborröhrchen (beispielsweise Zentrifugenröhrchen) vereint werden. Eine solche Trägerplatte dient dann der Aufnahme und dem Transport gleichartiger Proben, die denselben Präparationsschritt haben und demselben Präparationsprotokoll zu unterziehen sind. Der gemeinsame Speicherchip trägt unter anderem die gemeinsame Workflow-Definition[203]. Unter Verwendung dieser plattenbasierten Arbeitsweise lassen sich auch die in Kapitel 4.3 beschriebenen Lagerboxen für diskrete Substrate in die Bearbeitung durch den GHRC-Prototypen einbeziehen. Sie unterscheiden sich von den Trägerplatten für Standardröhrchen lediglich durch die zusätzliche Möglichkeit der selektiven Ansteuerung der diskreten Speicherchips der Substrate. Das Konzept der Trägerplatten ist hinsichtlich der Organisation des Systems in unabhängige Laborstationen unbedingt erforderlich. Während beim Testsystem der Scheduler ‚Bernstein' die einzelnen Prozesse der Workflow-Definition an die einzelnen Instanzen des ‚Generic Device' verteilt und ihre Ausführung synchronisiert hat, sind den autonomen Laborstationen des GHRC-Prototyps Workflow-Definition und aktueller Abarbeitungsstatus nicht bekannt. Diese Informationen liegen nur an der aktuell verwendeten Laborstation vor. Wenn die Abarbeitung des Workflows an einer anderen Station fortgesetzt werden soll, müssen Workflow-Definition und Workflowstatus zu dieser Station transportiert werden. Gleiches gilt auch für bis dahin akquiriertes Proben- und Pro-

---

[201] In diesem Zusammenhang handelsüblich sind Mikro- oder Nanotiterplatten, die auch beim Betrieb von HTS-Modulen verwendet werden.

[202] Die Verwendung eines gemeinsamen Speichermediums ist aus Performanzgründen im Zusammenhang mit der zügigen Abarbeitung von Proben relevant.

[203] Der gemeinsame Speicher einer Trägerplatte enthält unter anderem auch zusätzlich Informationen über die Trägerplatte selbst und über ihre Organisation (siehe auch die Datenstruktur für Cryo-Devices in Kapitel 5.6).

zesswissen und das Wissen über die zu bearbeitenden Proben. Die gemeinsamen Speicherchips der Trägerplatten werden zu diesem Zweck an der aktuellen Laborstation mit den aktuellen Informationen beschrieben (siehe dazu auch das Konzept des *Target-File* in Abschnitt 5.4.4) und stellen diese Informationen dann an einer nächsten Laborstation zur Verfügung. Das dortige Auslesen von Workflow-Definition, Workflowstatus und akquiriertem Wissen von den gemeinsamen Speicherchips der Trägerplatte erlaubt die Wiederaufnahme des Workflows an der richtigen Stelle (sofern Workflow-Beschreibungssprache und Workflow-Engine das Unterbrechen und die spätere Wiederaufnahme einer Workflow-Instanz erlauben, siehe auch Abschnitt 5.4.2). Die Platte dient somit der erforderlichen Synchronsierung zwischen den Laborstationen. Im Einzelnen:

- Nach Ausführung eines Bearbeitungsschritts an einer Station werden der erreichte Workflow-Status und das akquirierte Wissen auf die gemeinsamen Speicherchips der Trägerplatte geschrieben. Dies umfaßt auch alle entstandenen ID-Listen (siehe Abschnitt 5.4.4) und Zeitstempel, einschließlich der ToDo-Liste für den nächsten anstehenden Präparationsschritt.

- Eine Vorschau des unmittelbar nächsten Präparationsschritts muß an der aktuellen Station möglich sein. Dies erlaubt dem Anwender, die Eignung der aktuellen Station zu beurteilen und gegebenenfalls die Trägerplatte mit den Proben zu einer anderen, geeigneten Laborstation zu transportieren.

- Voraussetzung für die korrekte Wiederaufnahme einer Workflow-Instanz an der ausgewählten Laborstation ist das Einlesen von Workflow-Definition, Workflowstatus, dem bisher akquirierten Wissen und den bisherigen ID-Listen.

Für die Interaktion von GHRC-Prototyp[204] mit Trägerplatten für Standard-Laborröhrchen oder mit Lagerboxen wurden entsprechende Dockingstationen konzipiert. Eine integrierte Demultiplexerschaltung erlaubt den selektiven Zugriff auf die gemeinsamen Speicherchips von Trägerplatten oder Lagerboxen und zusätzlich auf die einzelnen Speicherchips diskreter Substrate. Die Speicherchips von Trägerplatten oder Lagerboxen können damit für die dezentrale Speicherung der Workflow-Definitionen und für die Dokumentationsakquise verwendet werden. Die Laborstationen können mit Hilfe einer geeigneten Ansteuerung (beschrieben in Abschnitt 5.5.2.2) mit den Dockingstationen und den Trägerplatten, Kryoboxen und diskreten Substraten interagieren. Folgende Abbildung zeigt eine Dockingstation mit aufgesetzter Lagerbox und aufgesetzter Trägerplatte für Standard-Laborröhrchen:

[204] Hier wird bewußt nicht nur Bezug auf die Stationensoftware genommen, da noch weitere Komponenten des GHRC-Prototyps diese Dockingstationen verwenden.

Abbildung 54: Links: eine Dockingstation mit aufgesetzter Lagerbox, die mehrere diskrete Substrate und ‚Portlinks' enthält. Rechts: eine Dockingstation mit aufgesetzter Trägerplatte für Standardröhrchen, in diesem Fall für 50ml Zentrifugenröhrchen. Die gemeinsamen Speicherchips der Trägerplatte ermöglichen eine Verwendung von Standard-Laborgefäßen mit dem GHRC-Prototypen. Die Dockingstation enthält eine Demultiplexerschaltung und erlaubt den selektiven Zugriff auf die gemeinsamen Speicherchips von Trägerplatten und Lagerboxen, aber auch den Zugriff auf die einzelnen Speicherchips von Substraten oder ‚Portlinks' in Lagerboxen.

5.4.4 Konzeption der Stationensoftware

Um eine Benutzerführung gemäß den in Kapitel 4.5 beschriebenen Prinzipien zu leisten und den in Abschnitt 5.3.3 identifizierten Anforderungen an den GHRC-Prototypen zu genügen, wird die Stationensoftware mit entsprechend benötigten Funktionalitäten ausgestattet und deshalb wie folgt konzipiert:

- Benutzerschnittstelle: Um die gewünschte Benutzerunterstützung bei manuellen oder semimanuellen Workflow-Aktivitäten zu leisten, enthält die Stationensoftware zusätzlich zu einer geeigneten Workflow-Engine (siehe Abschnitt 5.4.2) eine geeignete graphische Benutzerschnittstelle. Jede Instanz der Stationensoftware interagiert über ihr GUI mit dem Anwender. Anforderungen an eine geeignete Benutzerschnittstelle wurden im Rahmen einer Evaluierung des Testsystems identifiziert[205] und fließen in Abschnitt 5.4.6 in die Konzeption des GUI ein.

- Geräteintegration: Im Gegensatz zum Testsystem sollen vom GHRC-Prototypen auch Geräte reproduzierbar gesteuert und die von ihnen generierten Daten einheitlich und reproduzierbar akquiriert werden können. Daher müssen die Laborgeräte durch Abarbeitung entsprechender gerätebezogener Aktivitäten in der Worfklow-Definition gesteuert werden. Die Abarbeitung von Workflow-Aktivitäten erfolgt durch die lokale

[205] Gemäß der Intention des Testsystems als Grundlage für die Identifizierung von Anforderungen an Folgesysteme

Workflow-Engine (System Controller) der jeweiligen Laborstation. Die Interaktion der Laborgeräte muß dabei unter folgenden Randbedingungen erfolgen:

- Die Geräte sind den einzelnen Laborstationen zugeordnet.
- Jede Laborstation kann mit unterschiedlichen Laborgeräten ausgestattet sein. Dennoch soll die grundsätzlich gleiche Stationensoftware an allen Stationen verwendet werden.
- In verschiedenen Laboratorien stehen für denselben Protokollschritt unterschiedliche Laborgeräte zur Verfügung.
- Die Interaktion soll robust sein, d.h. der Ausfall eines Gerätes darf nicht zum Ausfall der gesamten Station führen.

Eine monolithische Architektur (siehe Abschnitt 3.4.2.1) ist dazu nicht geeignet. Benötigt wird eine modulare, flexible Anbindung der Laborgeräte an die Workflow-Engine über definierte, für alle Laborgeräte verwendbare Schnittstellen. Die dazu übliche Vorgehensweise ist die Verwendung von Device Integration Frameworks (siehe auch Abschnitt 3.4.2.2). Jede Instanz der Stationensoftware muß also neben der integrierten Workflow-Engine auch ein integriertes Device Integration Framework umfassen. Die Kommunikation zwischen Workflow-Engine und Device Integration Framework erfolgt mittels einer definierten, bidirektionalen Schnittstelle. Über diese Schnittstelle werden die benötigten Kommandos und Parameter aus den Aktivitäten an die peripheren Softwarekomponenten des Frameworks übertragen. Um mit unterschiedlichen konkreten Laborgeräten desselben Gerätetyps kompatibel zu sein[206], darf eine Geräte-Aktivität in der Workflow-Definition keine gerätespezifische Kommando- und Parametersyntax enthalten, sondern muß diese Kommandos und ihre Parameter von konkreten Geräten abstrahiert und in einer für die Geräte jeweils eines Gerätetyps gültigen Form repräsentieren. Dazu werden *Meta-Kommandos* und *Meta-Parameter* verwendet, die von der jeweiligen, gerätespezifischen peripheren Softwarekomponente auf Gerätelevel in konkrete Instruktionen für ein konkretes Laborgerät umgesetzt werden. Für jedes konkrete Laborgerät wird das Framework um eine solche modulare Applikation erweitert. In diesem Zusammenhang von Bedeutung ist das in dieser Arbeit so bezeichnete *GUI-Plugin-Konzept*: für jede periphere Softwarekomponente auf Gerätelevel wird ein *GUI-Plugin* entworfen, das beim Abarbeiten der Geräte-Aktivität im GUI der Stationensoftware angezeigt wird und mit dem Anwender während der Abarbeitung der Geräte-Aktivität interagiert. Die Laborstationen unterscheiden sich also nur hinsichtlich der vorhandenen modularen peripheren Geräteapplikationen. Für jeden Gerätetyp wird die Workflow-Beschreibungssprache

[206] Dies ist für den Austausch von Workflow-Definitionen zwischen Laboratorien relevant.

um eine instanziierbare und parametrierbare Meta-Aktivität erweitert, die gemeinsame Eigenschaften, Kommandos und Parameter der Geräte dieses Typs berücksichtigt[207].

- Proben- und Aliquot-Management: Die Korrelierbarkeit biologischer Effekte mit zeitlichen Aspekten (beispielsweise insgesamte Bearbeitungsdauer, Zeitaufwand für einen einzelnen Präparationsschritt, Dokumentierung von Unterbrechungen, usw.) ist im Besonderen bei zeitkritischen Bearbeitungsschritten oder bei sehr empfindlichen Proben bedeutsam, aber auch für eine lückenlose Dokumentation. Zu diesem Zweck verfügt die Stationensoftware über ein Proben- und Aliquot-Management, das einerseits für jeden Präparationsschritt die Anzahl der zu bearbeiten Proben und Aliquoten vorgibt, andererseits die Vollständigkeit der Abarbeitung überwacht und erfolgte Bearbeitungsschritte sekundengenau für jedes Aliquot dokumentiert. Auf diese Weise werden sowohl die Zeitpunkte der Probenbearbeitung dokumentiert, andererseits wird die Fehlpräparation von Proben oder ein Vergessen von Aliquots vermieden. Dazu wird das Zusammenspiel verschiedener ID-Listen benötigt, das im Folgenden erläutert wird.

    - Die in einer Präparation zu bearbeiten ProbenIDs und die Anzahl ihrer Aliquoten sind zu Beginn der Bearbeitung bekannt[208] und können in einer *initialen ToDo-Liste* zusammengefaßt werden. Faktisch handelt es sich dabei um eine Menge M:={(ProbenID, Anzahl)}, so daß M für jede zu bearbeitende ProbenID ein Tupel enthält, das die Anzahl der Aliquoten dieser ProbenID angibt. Die initiale ToDo-Liste ist faktisch die ToDo-Liste der ersten Workflow-Aktivität.

    - Die erfolgreiche Bearbeitung von Proben in einem Bearbeitungsschritt ist der Regelfall. Dennoch kann auch ein Verlust von Proben oder Aliquoten auftreten, beispielsweise in Folge einer Zentrifugation, so daß man grundsätzlich unterscheiden muß in

        - erfolgreich bearbeitete Aliquoten, die für den nächsten Bearbeitungsschritt zur Verfügung stehen (*Done-Liste*, D:={(ProbenID, Anzahl erfolgreich bearbeiteter Aliquoten)}),

        - verlorene Aliquoten, die für die weitere Bearbeitung nicht zur Verfügung stehen (*Lost-Liste*, L:={(ProbenID, Anzahl verlorener Aliquoten)}).

---

[207] Proprietäre Eigenschaften, die nicht allen Geräten dieser Klasse gemeinsam sind, werden von Meta-Activities somit nicht unterstützt.

[208] Dies ist der Regelfall. Für das Erstellen der intialen ToDo-Liste wird in Abschnitt 5.4.5 eine entsprechende Applikation konzipiert.

- Die ToDo-Liste einer Workflow-Aktivität ist also unmittelbar abhängig von der vorausgegangenen Aktivität. Somit kann keine einheitliche ToDo-Liste für alle Präparationsschritte gelten, vielmehr müssen für alle Aktivitäten jeweils eigene ToDo-, Done- und Lost-Listen geführt werden.

- Gemäß der Anforderung nach lückenloser Dokumentation und Erfassung der Bearbeitungsdauer ist in jedem Bearbeitungsschritt für jedes Aliquot der Zeitpunkt seiner abgeschlossenen Bearbeitung (erfolgreich oder nicht erfolgreich) zu erfassen. Die Stationensoftware dekrementiert dann die Anzahl der Aliquoten einer ProbenID in der ToDo-Liste der aktuellen Workflow-Aktivität und inkrementiert die Anzahl der Aliquoten der ProbenID entweder in der Done-Liste oder in der Lost-Liste des aktuellen Bearbeitungsschritts. Ein Bearbeitungsschritt ist dann abgeschlossen, wenn seine ToDo-Liste vollständig leer ist, d.h. wenn M={} gilt.

- Die Bearbeitung der nächsten anstehenden Aktivität setzt eine leere ToDo-Liste der aktuellen Aktivität voraus. Ein Erfassen nicht in der ToDo-Liste stehender Proben als ‚done' oder ‚lost' ist nicht möglich, so daß eine Fehlpräparation vermieden wird.

- Die Anzahl der Aliquoten einer ProbenID in der ToDo-Liste einer Aktivität teilt sich vollständig auf in die Done- und Lost-Listen der Aktivität, d.h. die Summe der Anzahl der Aliquoten einer ProbenID aus Done- und Lost-Liste ist gleich der Anzahl der Aliquoten der ToDo-Liste des Bearbeitungsschrittes.

- Die Done-Liste eines Präparationsschritts gibt Aufschluß über die Anzahl der erfolgreich bearbeiteten Aliquoten einer Probe, die Teil der ToDo-Liste des Schrittes gewesen sind. Ihr Inhalt ist aber nicht zwingend identisch mit der Anzahl der tatsächlich für den anstehenden Präparationsschritt vorhandenen Aliquoten, denn diese Anzahl kann durch Aliquotierung oder Pooling während eines Bearbeitungsschrittes in nicht vorhersagbarer Weise verändert werden[209]. Deswegen wird zusätzlich zur Done-Liste eine *Resulting*-Liste verwendet, die zunächst die gleichen Elemente enthält wie die Done-Liste, die aber mit Hilfe einer geeigneten Funktion[210] die nachträgliche Korrektur der Aliquotenanzahl der jeweiligen ProbenID durch den Anwender derart erlaubt, daß die Anzahl der Aliquoten der ToDo-Liste des anstehenden Schrittes der tatsächlichen im aktuellen Schritt resultierten Aliquotenanzahl entspricht.

---

[209] Siehe dazu auch die Beobachtungen bei der Ausführung eines Protokolls in Abschnitt 5.3.3.4.

[210] Diese Funktion greift in die Resulting-Liste eines Präparationsschritts ein und manipuliert die Aliquotenanzahl im Tupel einer ProbenID.

Daher gilt für alle Bearbeitungsschritte: die ToDo-Liste der nächsten Aktivität entspricht der Resulting-Liste der aktuellen Aktivität.

- Eine *Split-Aktivität* stellt einen Spezialfall dar. Sie erzeugt mehrere Resulting-Listen, jeweils eine für jeden von ihr ausgehenden Präparationspfad. Die Elemente der verschiedenen Resulting-Listen sind hinsichtlich ihrer Aliquotenanzahlen identisch mit den Elementen der ToDo-Liste, unterscheiden sich aber durch das vordefinierte Produkt-spezifische Suffix, das an die ProbenIDs der Elemente konkateniert wird. Eine Split-Aktivität wird beispielsweise nach einer Dichtezentrifugation eingesetzt, bei der die Proben vollständig in unterschiedliche Probenprodukte (beispielsweise Plasma und Sediment) aufgespaltet wurden, und erlaubt die Zuordnung der Probenprodukte zu unterschiedlichen folgenden Präparationspfaden (siehe dazu auch Abschnitt 2.2.3). Die jeweilige Resulting-Liste ist die initiale ToDo-Liste des jeweiligen anschließenden Pfades.

- *Label-on-Demand*: durch die in Abschnitt 5.3.3.4 beschriebene Videoanalyse ‚impliziter' Anforderungen an den GHRC-Prorotypen wurde deutlich, daß Proben, Aliquoten oder Probenprodukte während einer Präparation verschiedene Probengefäße (beispielsweise Zentrifugenröhrchen, Zellkulturflaschen oder Kryoröhrchen) durchlaufen. Der Inhalt eines Probengefäßes muß jederzeit identifizierbar sein. Daher muß ein Transfer der jeweiligen Beschriftung[211] erfolgen. Dieser Transfer muß sehr flexibel möglich sein, denn es ist nicht exakt absehbar, wieviele Probengefäße für eine Probe oder ihre Aliquoten während einer Präparation benötigt werden. Um den Anwender effizient zu unterstützen, erlaubt die Stationensoftware das Duplizieren von Etiketten bei Bedarf, unter Verwendung eines Barcodescanners und eines Barcodedruckers. Diese Geräte sind modular mittels Meta-Applikationen an die Stationensoftware angebunden und interagieren mittels Meta-Befehlen, die die Verwendung verschiedener Gerätemodelle erlauben.

- Blättern und Vorschaufunktion: die Eignung einer Laborstation für die Abarbeitung bestimmter Präparationsschritte wird bestimmt durch die Laborgeräte der Station. In der Regel erfordern die verschiedenen Bearbeitungsschritte eines Protokolls (also die Aktivitäten einer Workflow-Definition) die Nutzung mehrerer verschiedener Laborstationen hintereinander. Die benötigten Laborstationen sind nicht in der Workflow-Definition vordefiniert, sondern der Anwender sucht die jeweils passende Laborstation aus, um einen bestimmten anstehenden Protokoll-

[211] Die gegenwärtige Laborpraxis besteht in einer handschriftlichen Kennzeichnung der Röhrchen oder im Aufbringen vorab ausgedruckter Etiketten auf die Probengefäße. Nachteile: (1) Es muß von vornherein klar sein, welche Proben im Labor präpariert werden, (2) die Anzahl der vorbereiteten Etiketten kann entweder zu hoch oder zu niedrig sein, (3) der Anwender muß das benötigte vorbereitete Etikett zunächst suchen.

schritt auszuführen. Dazu ist es notwendig, daß der Anwender mindestens den jeweils nächsten anstehenden Präparationsschritt einsehen kann. Dies ist gewährleistet, denn die Stationensoftware kann grundsätzlich jeden Präparationsschritt abbilden[212] und somit den Anwender auch über den jeweils nächsten anstehenden Präparationsschritt informieren[213], selbst wenn der Schritt an der Station nicht ausgeführt werden kann.

- *Target-File*: Gemäß den Ausführungen in Abschnitt 5.4.3 sind die gemeinsamen Speicherchips von Trägerplatten und Lagerboxen erforderlich für den Austausch von Workflow-Definition, Abarbeitungsstatus, ID-Listen-Management und akquiriertem Proben- und Prozeßwissen zwischen den einzelnen Laborstationen. Die ID-Listen, der Status der Workflow-Instanz und das akquirierte Wissen werden zu diesem Zweck sukzessive in einer gemeinsamen[214] Datei, deren Struktur sich an der Struktur der Worfklow-Definition orientiert, gesammelt und aktualisiert. Dieses *Target-File* enthält für jede Aktivität (in diesem Zusammenhang auch als *Target* bezeichnet) der Workflow-Definition eine zunächst leere spezifische Struktur, in die das Wissen eingetragen wird, das bei Ausführung der Aktivität generiert wird. Das Target-File stellt auch die Wissensbasis für die Extraktion eines Laborreports dar. Um auf Basis des Target-Files ein *Präparationstracking* durch eine Probendatenbank zu ermöglichen, muß das Target-File unabhängig von proprietären Datenbankformaten und unabhängig von proprietären Formaten der spezifisch verwendeten Workflow-Engine sein. Hinsichtlich potentiell langer Lagerzeiten und der eventuell anschließenden Wiederaufnahme einer Workflow-Instanz ist für das Target-File ein langzeitkompatibles Format erforderlich. Basierend auf den Ausführungen in Abschnitt 5.2.2 wurde auch hier XML gewählt. Die Generierung des zu einer Workflow-Definition gehörenden Target-File-Templates wird im Zusammenhang mit dem Workflow-Editor in Abschnitt 5.5.1.4 beschrieben, die Instanziierung im Zusammenhang mit dem Erstellen einer intialen ID-Liste in Abschnitt 5.4.5.

- Spezifische Adaption der Stationensoftware an Lagerboxen und Trägerplatten: je nach verwendeter Lagerbox oder Trägerplatte wird die Darstellung ihres Layouts in der Benutzerschnittstelle der Stationensoftware angepaßt und der Zugriff auf

---

[212] Dies resultiert aus der Verwendung derselben Stationensoftware an allen Stationen.

[213] Dies entspricht der Anforderung nach einer Vorschau aller Protokollschritte, die von den Anwendern formuliert wurde. Voraussetzung ist eine Benutzerschnittstelle, die ein solches Blättern durch die Aktivitäten einer Workflow-Definition darstellen kann.

[214] Die Speicherung in einer gemeinsamen Datei wird dadurch motiviert, dass die Informationen nur zusammen einen Mehrwert ergeben. Voneinander isoliert sind die Informationen wie beispeilsweise ID-Listen oder Zeitstempel wertlos.

die Speicherchips gesteuert[215]. Dies erfolgt auf Basis der Informationen, die unter Verwendung der Formatierungsapplikation (siehe Abschnitt 5.4.5) in den gemeinsamen Speicherchips der verschiedenen Cryo-Devices (siehe Kapitel 4.3) eingetragen werden (siehe auch Abschnitt 5.5.3.1).

5.4.5 Konzeption weiterer Systemkomponenten

Zusätzlich zur Stationensoftware sind verschiedene weitere Applikationen erforderlich, deren Verwendung der Abarbeitung von Workflow-Instanzen vorausgehen oder sich daran anschließen. Ihr Funktionsumfang wird im Folgenden grob beschrieben.

- Formatierungsapplikation: Um mit der elektronischen Lagertank-Infrastruktur (siehe Kapitel 4.4) und mit dem GHRC-Prototypen interagieren zu können, muß jedes Cryo-Device[216] entsprechend formatiert sein. Das bedeutet, daß die gemäß Kapitel 5.6 definierten Datenstrukturen auf den Speicherchips vorhanden sein und die relevanten Informationen[217] über das konkrete Cryo-Device enthalten müssen. Im Zusammenhang mit dem GHRC-Prototypen sind folgende Cryo-Devices unmittelbar von Belang: Trägerplatten für verschiedene Standard-Laborgefäße, Lagerboxen, diskrete Substrate (‚Icebreaker') und ‚Portlinks' zur Adaption von Standard-Kryoröhrchen. Die Formatierungsapplikation dient dem Aufspielen in Kapitel 5.6 konzipierten Datenstrukturen auf das jeweilige Cryo-Device und dem Eintragen der relevanten Informationen über das jeweilige Cryo-Device.

- Initialisierungsapplikation: die Stationensoftware interagiert mit den Informationen auf den gemeinsamen Speicherchips von Trägerplatten oder Lagerboxen. Voraussetzung ist, daß Workflow-Definition, initiale ToDo-Liste und instanziiertes Target-File auf den Speicherchips vorhanden sind. Die Initialisierungsapplikation dient der Auswahl einer Workflow-Definition, dem Erstellen der initialen ToDo-Liste der zu präparierenden Proben und Aliquoten (siehe Abschnitt 5.4.4), dem Instanziieren des Target-Files[218] und dem anschließenden Speichern auf den Speicherchips des jeweiligen Cryo-Devices. Abhängig von der Formatierung des verwendeten Cryo-Devices nutzt

---

[215] Bei Trägerplatten werden ausschließlich die gemeinsamen Speicherchips, bei Lagerboxen zusätzlich die Speicherchips der diskreten Röhrchen verwendet.

[216] Trägerplatten, Lagerboxen und intelligente Kryoröhrchen sind Cryo-Devices im Sinne von Kapitel 4.3. Ihre Kompatibilität mit der elektronischen Lagertank-Infrastruktur wird durch die in Kapitel 5.6 beschriebenen Datenstrukturen erreicht.

[217] In Bezug auf Trägerplatten oder Lagerboxen sind dies Informationen über Geometrie, Anzahl und Volumen der Steckplätze und deren Kompatibilität mit Standard-Röhrchen oder ‚Icebreakern' oder ‚Portlinks'.

[218] Die initiale ToDo-Liste wird in einem Protokoll-spezifischen Target-File gespeichert.

die Initialisierungsapplikation nur die gemeinsamen Speicherchips einer Trägerplatte oder auch die einzelnen Speicherchips von Substraten oder ‚Portlinks'.

- Ansichts- und Exportapplikation: Um die Informationen der gemeinsamen Speicherchips von Trägerplatten und Lagerboxen und die Informationen der einzelnen Speicherchips von ‚Icebreakern' oder ‚Portlinks' beispielsweise beim Probenaustausch als unmittelbar lesbare Dokumente bereitzustellen, ist der Export der Informationen auf externe Datenträger (beispielsweise USB-Sticks) erforderlich. Zu diesem Zweck wird eine Applikation benötigt, die das selektive Auslesen von Speicherchips erlaubt und den Inhalt darstellen sowie auf ein externes Speichermedium exportieren kann.

- Applikation zum Erstellen von Workflow-Definitionen und Target-Files: Zum Erstellen von Workflow-Definitionen in der gewählten Workflow-Beschreibungssprache ist ein Editor erforderlich, der Aktivitäten gemäß den Ausführungen in Abschnitt 5.4.2 zur Verfügung stellt und zusätzlich zur Workflow-Definition automatisch ein korrespondierendes ‚leeres' *Target-File-Template* generiert (siehe Abschnitt 5.5.1.4).

5.4.6 Ein GUI für die Stationensoftware des GHRC-Prototyps

In Abschnitt 5.3.3.1 wurden funktionale Anforderungen an den GHRC-Prototypen beschrieben, die anhand des Testsystems identifiziert worden waren. Auch hinsichtlich ergonomischer Aspekte kann das Testsystem als Grundlage für die Identifizierung von Anforderungen verwendet werden, die bei der Konzeption einer Benutzerschnittstelle relevant sind.

Die Mensch-System-Interaktion zwischen Anwendern und Testsystem wurde im Rahmen eines praktischen Anwendungstests untersucht. Die Eignung der Benutzerschnittstellen aus Sicht der Anwender wurde mit Hilfe von Fragebogen in Anlehnung an DIN EN ISO 9241 und in freiem Gespräch evaluiert. Gegenstand der Evaluierung waren die beiden Systemkomponenten ‚Bernstein' und ‚Generic Device'. Untersucht wurden jeweils:

- Selbstbeschreibungsfähigkeit: „Gibt Ihnen die Software genügend Erläuterungen und ist sie in ausreichendem Maße verständlich?"

- Erwartungskonformität: „Kommt die Software durch eine einheitliche und verständliche Gestaltung Ihren Erwartungen und Gewohnheiten entgegen?"

- Fehlertoleranz: „Bietet die Software die Möglichkeit, trotz fehlerhafter Eingaben das beabsichtigte Arbeitsergebnis ohne oder mit geringem Korrekturaufwand zu erreichen?"

- Lernförderlichkeit: „Ist die Software so gestaltet, daß Sie sich ohne großen Aufwand in sie einarbeiten konnten und bietet sie auch dann Unterstützung, wenn Sie neue Funktionen lernen möchten?"

5.4.6.1 Evaluierung des GUI von ‚Bernstein'

Zur Evaluierung des GUI von ‚Bernstein' wurden insgesamt 17 Fragen zu den oben genannten Aspekten gestellt. Die Testpersonen sollten mit Werten auf einer Skala von -3 bis +3 antworten. Bester möglicher Wert für jede Frage war der Wert +3. Jede Frage wurde mit einem durchschnittlichen Wert deutlich unter 0 beantwortet. Niedrigster Durchschnittswert war -3, höchster Durchschnittswert war -0,75. Der Gesamtdurchschnitt beträgt -2,1. ‚Bernstein' wurde von den Testpersonen als unübersichtlich und als nicht intuitiv bedienbar beurteilt. Ein Überblick über das Funktionsangebot wurde durch die Verwendung von Begriffen, Bezeichnungen, Abkürzungen und Symbolen im GUI erschwert, die den Testpersonen schwer oder gar nicht verständlich waren. Das Funktionsangebot konnte auch nicht durch eine Hilfefunktion vermittelt werden, da eine solche Funktion nicht vorhanden ist. Somit waren auch situationsspezifische Erläuterungen nicht verfügbar. Gestaltung und Bedienkonzept von ‚Bernstein' wurden als deutlich von anderen (Standard-) Applikationen abweichend empfunden, so daß den Testpersonen eine erfahrungsgeleitete Bedienung nicht möglich war. Folgende Abbildung zeigt das GUI von ‚Bernstein' und beschreibt seine Elemente:

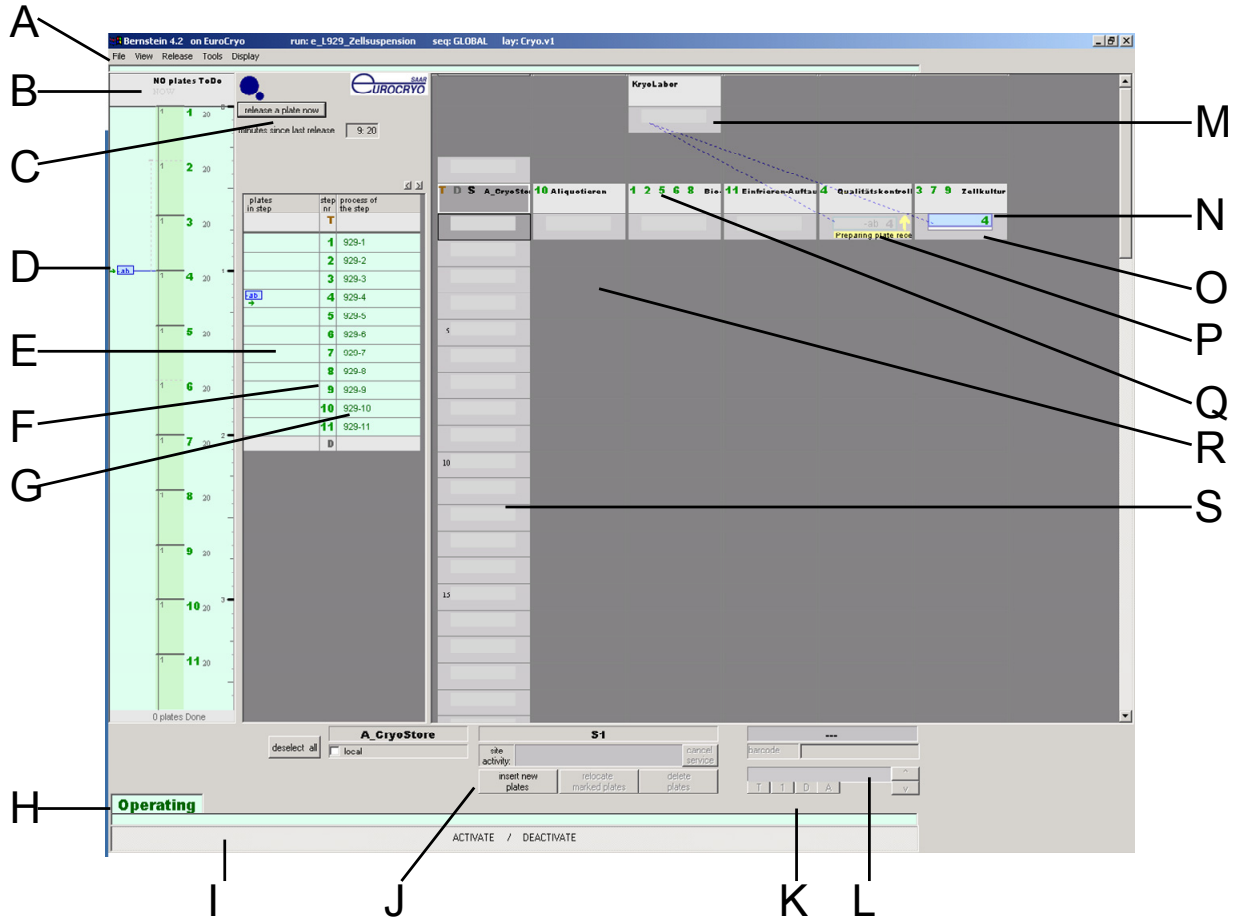

Abbildung 55: Die graphische Benutzerschnittstelle des Schedulers ‚Bernstein'. (A) Hauptmenü mit den Punkten ‚File', ‚View', ‚Release', ‚Tools' und ‚Display'; (B) Statusmeldung über die Anzahl der abzuarbeitenden Probenträger; (C) Button zum manuellen Start der Bearbeitung eines ausgewählten Probenträgers; (D) Anzeige der aktuellen Prozess-Aktivität für einen Probenträger innerhalb des aktuellen Prozesses; (E) Übersicht der einem

Prozeß aktuell zugeordneten Probenträger; (F) Prozeßliste des Workflows; (G) Namen der Prozesse in der ‚Evoscreen'-Datenbank; (H) Betriebsanzeige des Schedulers; (I) Activate/Deactivate-Button zum Starten oder Anhalten der Synchronisiation durch den Scheduler; (J) Button zum Einfügen neuer Probenträger in eine ToDo-Liste und weitere Buttons zum Löschen und Verschieben von zu bearbeitenden Probenträgern innerhalb der ToDo-Liste; (K) Buttons zum manuellen Setzen eines bestimmten Prozesses für einen zu bearbeitenden Probenträger; (L) Anzeigefeld für eine Probenträger-ID; (M) Anzeigeelement für einen Probentransport, repräsentiert die Kapazität eines Roboters oder einer Person und durch gestrichelte Linien einen anstehenden Transport zwischen verschiedenen Geräten oder zwischen Laborstationen; hier steht nun ein Probentransport von der Station ‚Zellkultur' zur Station ‚Qualitätskontrolle' an; (N) das blaue Rechteck symbolisiert einen Probenträger in Bearbeitung. Die grüne Ziffer 4 sagt aus, dass nun Prozeß 4 aktuell ist; (O) die grauen Rechtecke in dieser Zeile symbolisieren die verschiedenen Laborstationen (‚Aliquotieren', ‚Biophysikalische Manipulation', ‚Einfrieren/Auftauen', ‚Qualitätskontrolle' und ‚Zellkultur'); (P) Meldung, dass die Station zum Empfang eines Probenträgers bereit ist; (Q) die grünen Ziffern repräsentieren die Nummern der Prozesse, die im aktuellen Workflow der jeweiligen Station zugeordnet sind; diesen Nummern sind in (G) die Prozeßnamen zugeordnet; (R) Bereich für weitere Laborstationen; (S) symbolisches Lager für anstehende und bereits bearbeitete Probenträger.

Das Fehlen geeigneter farblicher Akzente als Orientierungshilfe wurde beanstandet, ebenso die schwierige Orientierung durch unklare Aufteilung des Bildschirms, durch eine Fülle von Bedienelementen mit unklarer Funktionalität und zusätzlich durch deren Anordnung (Beispiel: Form und Position der ‚Activate/Deactivate'-Schaltfläche). In diesem Zusammenhang wurden auch Darstellungsfehler und zu klein gewählte Schrift bemängelt, die zusätzlich die Orientierung behindert haben. In Abhängigkeit der Anzahl von Laborstationen ist eine vollständige Darstellung der vorhandenen Laborstationen nicht möglich. Die Testpersonen konnten weder anhand des GUI noch anhand von Betriebsmeldungen den Betriebszustand oder die Aktivitäten der Applikation nachvollziehen. Aus den genannten Schwierigkeiten resultierte, daß es den meisten Testpersonen nicht gelungen ist, selbständig einen Workflow auszuwählen und zu starten. Der Korrekturaufwand im Fehlerfall wurde als sehr hoch eingestuft. Fehlermeldungen wurden als schlecht verständlich beurteilt, Hinweise zur Fehlerbehebung wurden vermißt. Die Testpersonen haben im Fehlerfall einen Neustart der Applikation oder des Computers durchgeführt. Die Testpersonen haben die Lernförderlichkeit von ‚Bernstein' als sehr niedrig beurteilt, ebenso die Erlernbarkeit der Applikation ohne fremde Hilfe oder Handbuch. Der erforderliche Zeitaufwand zum Erlernen der Bedienung wurde als sehr hoch eingestuft, gleichzeitig wurde die Notwendigkeit des Einprägens vieler Details bemängelt. Die Gestaltung von ‚Bernstein' wurde jedoch als nicht geeignet beurteilt, sich einmal gelernte Dinge einzuprägen, und hat zudem nicht zum Ausprobieren von Funktionen ermutigt.

5.4.6.2 Evaluierung des GUI von ‚Generic Device'

Das GUI von ‚Generic Device' ist in Abbildung 37 illustriert und beschrieben. Es wurde hinsichtlich der oben genannten Aspekte mit Hilfe von 20 Fragen evaluiert. Die Testpersonen sollten mit Werten auf einer Skala von -3 bis +3 antworten. Bester möglicher Wert für jede Frage war auch hier der Wert +3. Von den 20 Fragen wurden 9 mit einem durchschnittli-

chen Wert unter 0 beantwortet, 11 wurden mit einem durchschnittlichen Wert über 0 beantwortet. Niedrigster Durchschnittswert war -2,6, höchster Durchschnittswert war +2,5. Der Gesamtdurchschnitt beträgt +0,0175. Damit wurde das GUI von ‚Generic Device' deutlich besser bewertet als das GUI von ‚Bernstein', auch wenn die Differenz zwischen dem Gesamtdurchschnitt und dem möglichen Bestwert einen deutlichen Verbesserungsbedarf aufzeigt. ‚Generic Device' wurde von den Testpersonen als wesentlich übersichtlicher als ‚Bernstein' empfunden. Zwar stellt auch diese Applikation mangels Hilfefunktion weder auf Anforderung noch von sich aus situationsspezifische Erklärungen zur Verfügung, wurde aber dennoch hinsichtlich des Überblicks über ihr Funktionsangebot deutlich besser beurteilt als ‚Bernstein'. Die verwendeten Begriffe, Bezeichnungen und Symbole im GUI waren den Benutzern hinreichend verständlich und wurden teilweise als sehr gut bewertet. Gleiches gilt für die Unterteilung des GUI in *Ausgabebereich*, *Eingabebereich* und *Statusbereich*, die den Testpersonen die Orientierung erleichtert hat, auch wenn die Größe der jeweiligen Bereiche von den einzelnen Testpersonen heterogen bewertet wurde. Tendenziell wurde der Ausgabebereich hinsichtlich des Umfangs darzustellender Instruktionen als zu klein empfunden, während der Eingabebereich für die Dokumentationsakquise hinsichtlich der Anzahl und Größe der auszufüllenden Formularfelder als zu groß empfunden wurde. Schriftgröße und Erkennbarkeit im Ausgabebereich wurden von allen Testpersonen als sehr gut beurteilt, im Eingabebereich jedoch eher als mittelmäßig. Der Statusbereich wurde von den Testpersonen bemängelt insofern, als er wegen ineffizienter Platzierung von Anzeigeelementen und Logos zu groß sei und daher Ausgabebereich und Eingabebereich nach unten verdränge. Die Verständlichkeit der Bedien- und Anzeigeelemente des Statusbereichs hingegen wurde positiv beurteilt, wenn auch einige Informationen als für den Anwender nicht relevant eingestuft wurden. Die Darstellung einiger für die Präparation wichtiger Informationen (beispielsweise ProbenID) des Statusbereichs wurde jedoch als zu klein empfunden. Die nicht vorhandene Fehlertoleranz wurde bemängelt, denn eine Korrekturmöglichkeit fehlerhafter Eingaben nach irrtümlichem Abschließen eines Präparationsschrittes ist nicht gegeben[219], und Warnungen bei fehlerhaften Eingaben oder bei irrtümlichem Abschließen einer Aktivität fehlen[220]. Diese Erfahrungen zusammen mit der Anforderung der Anwender nach flexiblen Dokumentationszeitpunkten[221] haben das weiter unten beschriebene Konzept für die flexible und korrigierbare Akquise von Dokumentationsdaten motiviert. Die Benutzerschnittstelle von ‚Generic Device' wurde hinsichtlich der Lernförderlichkeit sehr positiv beurteilt. Aus Sicht der Testpersonen erfordert sie nur wenig Einarbeitungszeit, ermutigt zum Ausprobieren von Funktionen, ist so gestaltet, daß

[219] Siehe auch Abschnitt 5.3.2. Dies war Resultat der wechselweisen, sequentiellen Definition von Befehlen zur Anzeige von Instruktionen an den Anwender und zum Verarbeiten von Eingabeformularen.

[220] Im GHRC-Prototyp wird ein irrtümliches Abschließen einer Aktivität durch das konzipierte Proben- und Aliquoten-Management verhindert.

[221] Siehe auch Abschnitt 5.3.2.

sich einmal Erlerntes gut einprägt, erfordert nicht, daß man sich viele Details merken muß und ist einfach ohne fremde Hilfe oder Handbuch erlernbar. Das GUI von ‚Generic Device' entspricht insgesamt den Erwartungen und Gewohnheiten der Testpersonen in einem wesentlich höheren Maße als das GUI von ‚Bernstein', insbesondere auch hinsichtlich des Feedbacks an den Benutzer und eines einheitlichen Bedienprinzips.

Als problematisch wurde von den Testpersonen jedoch die vollkommen uneinheitliche Gestaltung der Benutzerschnittstellen der beiden Applikationen ‚Bernstein' und ‚Generic-Device' empfunden, aus denen sich das Testsystem zusammensetzt. Die daraus resultierenden uneinheitlichen Bedienkonzepte innerhalb des Systems haben die Bedienung insgesamt trotz der deutlich besseren Beurteilung des GUI von ‚Generic Device' erheblich erschwert.

5.4.6.3 Konzeption eines GUI für die Stationensoftware des GHRC-Prototyps

Die Konzeption der graphischen Benutzerschnittstelle für den GHRC-Prototypen ergibt sich einerseits aus der Konzeption der Stationensoftware[222] und andererseits aus den oben beschriebenen Evaluierungsergebnissen.

Wie in Abschnitt 5.4.1 im Zusammenhang mit dem konzipierten Aufbau des Systems aus autonomen Laborstationen beschrieben, muß die jeweils aktuell verwendete Laborstation eine Vorschau mindestens des nächsten anstehenden Präparationsschrittes leisten können. Dadurch erhält der Anwender die Möglichkeit, eine für die weitere Abarbeitung des Workflows geeignete andere Laborstation auszuwählen[223] oder an der aktuellen Station zu verbleiben. Die Konzeption des GHRC-Prototyps unterstützt eine solche Vorschau grundsätzlich, da sie die vollständige Workflow-Definition auf den gemeinsamen Speicherchips derjenigen Trägerplatte oder Lagerbox vorsieht, auf der die zu bearbeitenden Proben und Aliquoten organisiert sind, so daß die Workflow-Definition an der jeweils aktuellen Station vollständig vorhanden ist. Die in der Konzeption der Stationensoftware vorgesehene Workflow-Engine erlaubt die vorschauende Abbildung der nächsten anstehenden Workflow-Aktivität. Demnach muß das GUI der Stationensoftware so konzipiert werden, daß es eine solche Vorschau abbilden kann. Aus Anwendersicht wurde in Abschnitt 5.3.3.2 zusätzlich als Anforderung an die Stationensoftware genannt, eine Gesamtübersicht des vollständigen Protokolls zu ermöglichen. Als Orientierungshilfe und für das Vorbereiten der einzelnen Präparationsschritte wollen die Anwender die Möglichkeit haben, jederzeit zu jedem gewünschten vorherigen oder nachfolgenden Präparationsschritt zu blättern. Eine solche beliebige Navigation durch die einzelnen Schritte eines Protokolls geht deutlich über die Anforderung nach einer Vorschau des

---

[222] siehe Abschnitt 5.4.4.

[223] Dies entspricht der in Abschnitt 5.3.3.1 beschriebenen Anforderung, wonach die Auswahl von Laborstationen durch Benutzer erfolgen soll.

nächsten Schrittes hinaus, basiert aber prinzipiell auf den gleichen Mechanismen. Es muß lediglich eine Darstellungsmethodik gewählt werden, mit deren Hilfe das GUI der Stationensoftware ein solches Blättern effizient und übersichtlich darstellen kann. Eine geeignete Methodik ist die Darstellung der Workflow-Definition in Registerform, bei der jeder einzelne Protokollschritt (d.h. jede einzelne Workflow-Aktivität) in Form einer eigenen Registerkarte dargestellt wird. Die verschiedenen Protokollschritte können dann durch Auswahl der jeweiligen Registerkarten angezeigt werden, und ein Blättern zu jedem beliebigen Protokollschritt kann durch entsprechende Bedienelemente ermöglicht werden. Diese Form der Darstellung ermöglicht eine prinzipiell einheitliche Darstellung aller Aktivitäten als Registerkarten und eine einheitliche Darstellung aller Workflow-Definitionen in Form von Registeranordnungen. Auch die in Abschnitt 5.3.3.2 formulierte Anforderung nach vollständigen Material- und Reagenzienlisten[224], die zu jedem Zeitpunkt der Präparation wieder eingesehen werden können sollen, läßt sich durch das Konzept der Registerkarten umsetzen und in die Workflow-Definition integrieren. Das Bereitstellen von Materialien und Reagenzien kann dann sogar als eigene Aktivitäten in der Workflow-Definition als Voraussetzung für die Abarbeitung der eigentlichen Protokollschritte definiert und mit Zeitstempel in der Dokumentation erfaßt werden. Um eine effiziente Orientierung zwischen den einzelnen Registerkarten zu erlauben, besonders aber um zwischen dem aktuellen Protokollschritt und später noch folgenden bzw. bereits abgearbeiteten Protokollschritten mit Hilfe des GUI zu unterscheiden, wurde die Verwendung von Farben und Farbkombinationen[225] für die entsprechenden Registerkarten konzipiert. Gleiches gilt für die Unterscheidung von aktiven und inaktiven Bedienelementen des GUI.

In Bezug auf die effiziente Orientierung der Anwender und hinsichtlich einer intuitiven Systembedienung sind außerdem die beschriebenen Ergebnisse der Evaluierung (siehe Abschnitte 5.4.6.1 und 5.4.6.2) relevant. Die Evaluierung des GUI von ‚Generic Device' hat gezeigt, daß eine klare Aufteilung in verschiedene logische Bereiche (Ausgabebereich, Eingabebereich und Statusbereich) diesbezüglich vorteilhaft ist. Deshalb wird eine solche Unterteilung auch für die konzipierte Registeranordnung des GHRC-Prototyps konzipiert. Allerdings sind hierbei die in Abschnitt 5.3.3.2 von Anwendern formulierten Anforderungen nach (1) flexibler Datenakquise[226] und nach (2) einem jederzeit verfügbaren Überblick über alle zu akquirierenden und akquirierten Werte der Dokumentation zu berücksichtigen, ebenso die Anforderung nach Fehlertoleranz und einer Korrekturmöglichkeit für fehlerhaft eingegebene Daten. Daher muß die Datenakquise von Protokollschritten entkoppelt stattfinden. Sie darf nicht

---

[224] Die Anwender wünschen bereits vor Präparationsbeginn eine vollständige Übersicht über die benötigten Materialien und Reagenzien, damit spätere Arbeitsunterbrechungen für das Bereitstellen der Materialien vermieden werden. Ein Einblick in diese Listen soll dem Anwender jederzeit möglich sein.

[225] Die Wirkung von Farbkombinationen auf den Anwender und ihre besondere Bedeutung bei der Konzeption von Benutzerschnittstellen wird beispielsweise in [Laugwitz 2001] untersucht.

[226] Der Anwender will selbst entscheiden, wann er Daten erfaßt.

mehr durch Eingabeformulare erfolgen, die nach Bearbeitung keine Datenkorrektur mehr zulassen und deren Verarbeitung erst die Voraussetzung für den nächsten Protokollschritt ist. Zusätzlich hat die Evaluierung des GUI von ‚Bernstein' gezeigt, daß eine Fülle von Bedienelementen die Selbstbeschreibungsfähigkeit einer Applikation offensichtlich vermindert und die Orientierung des Anwenders deutlich erschwert. Um den Anwender bei der Probenbearbeitung sinnvoll zu unterstützen, sollten daher nicht benötigte Anzeige- und Bedienelemente vermieden und das GUI auf die erforderlichen Elemente reduziert werden. Dies motiviert zusammen mit der oben beschriebenen Entkopplung der Datenakquise den Verzicht auf einen eigenen Eingabebereich im GUI. Stattdessen wird im GUI eine Schaltfläche konzipiert, die das beliebige Ein- und Ausblenden eines Dokumentationsditors ermöglicht. Dieser bietet eine Übersicht aller zu akquirierenden Dokumentationsdaten[227] und erlaubt gleichzeitig ihre Akquise und Korrektur, zu jedem beliebigen Zeitpunkt der Präparation. Der Dokumentationseditor erlaubt aber auch die Eingabe beliebiger Texte, um zusätzliche Daten oder Abweichungen vom Protokoll zu dokumentieren. Dadurch wird (1) die gewünschte zeitlich flexible Datenakquise ermöglicht, unabhängig vom aktuellen Bearbeitungsschritt, (2) die Möglichkeit der Korrektur falsch akquirierter Daten geschaffen und (3) die Erfassung von Abweichungen vom Protokoll möglich. Das Bestätigen der Vollendung eines Präparationsschrittes kann nun mit Hilfe einer eigenen Schaltfläche erfolgen und ist nicht mehr abhängig von zugeordneten Formularen.

Die Aufteilung des GUI der Stationensoftware erfolgt somit nur in Ausgabebereich und Statusbereich. Dadurch kann der Ausgabebereich entsprechend vergrößert werden, und der Statusbereich kann anders platziert werden. Beides entspricht den Anforderungen, die bei der Evaluierung des GUI von ‚Generic Device' deutlich wurden. Der Ausgabebereich dient zum Anzeigen von Instruktionen an den Benutzer, aber auch zum Darstellen der GUI-Plugins der Meta-Applikationen (siehe auch Abschnitt 5.4.4). Die Konzeption des Systems aus autonomen Laborstationen macht diejenigen Statuselemente, die bei ‚Generic Device' den Verbindungsstatus mit ‚Bernstein' angezeigt haben, überflüssig. Der Fortschritt der Präparation wird durch die Registeranordnung und durch das Benennen der Registerzungen deutlich, so daß auf Fortschrittsbalken und Schrittbezeichnungen im Statusbereich verzichtet werden kann. Der Statusbereich wird stattdessen unter anderem für das konzipierte ID-Management (siehe Abschnitt 5.4.4) verwendet. Für jeden Präparationsschritt zeigt der Statusbereich die Übersicht über die Proben in der ToDo-, Done-, Lost- und Resulting-Liste an und erlaubt eine Korrektur der Aliquotenanzahlen. Deshalb wird der Statusbereich ebenso wie der Ausgabebereich auf den Registerkarten selbst angeordnet.

---

[227] Die Menge des zu akquirierenden Wissens soll daher in der Workflow-Definition in Form einer entsprechenden Aktivität definiert werden können (siehe dazu auch Abschnitt 5.5.1.4).

Den obigen Ausführungen zur Reduzierung der Anzeigeelemente im GUI folgend, wird eine Adaption des GUI an den jeweiligen Präparationsschritt konzipiert insofern, als nur die potentiell benötigten Anzeigeelemente tatsächlich dargestellt werden. Dazu zählen je nach Präparationsschritt beispielsweise Schaltflächen zum Registrieren von Reagenzien, zum Duplizieren von Etiketten oder zum Registrieren des Bearbeitungszustandes von Proben. Entsprechendes Aktivieren und Deaktivieren von GUI-Elementen ist in den jeweiligen eigenentworfenen Aktivitäten (siehe Abschnitt 5.5.1.4) vorgesehen.

Das GUI soll jederzeit, unabhängig von Präparationsschritten, Bedienelemente für folgende Funktionen anzeigen:

- Einlesen einer Trägerplatte[228]
- Beschreiben einer Trägerplatte[229]
- Aufrufen einer Hilfefunktion
- Generieren eines Reports[230]
- Akquise schwierig verbalisierbaren Wissens

Die Konzeption des GUI sieht außerdem die Bedienung der Elemente per Touchscreen vor und erfordert eine entsprechende Platzierung und Größe der Elemente.

Die erarbeitete Konzeption war die Grundlage für das konkrete Design[231] der graphischen Benutzerschnittstelle für die Stationensoftware des GHRC-Prototyps. Dieses Design wird in Abschnitt 5.5.3.3 im Zusammenhang mit der Implementierung der Stationensoftware ausführlich illustriert und beschrieben.

Die Bedienung des Testsystems wurde durch die uneinheitliche Gestaltung und die unterschiedlichen Bedienkonzepte der GUIs der beiden Applikationen ‚Bernstein' und ‚Generic Device' verkompliziert und erschwert. Um ähnliche Probleme beim GHRC-Prototypen zu vermeiden, wird das oben konzipierte GUI auch für die weiteren, in Abschnitt 5.4.5 konzipierten Systemkomponenten des GHRC-Prototyps hinsichtlich seines prinzipiellen Erscheinungsbildes und Bedienkonzeptes beibehalten, aber jeweils funktional angepaßt.

---

[228] Beim Ankommmen einer Trägerplatte oder einer Lagerbox.

[229] Unmittelbar vor dem Transport einer Trägerplatte oder Lagerbox zu einer anderen Laborstation.

[230] Auf Basis der Informationen des Target-Files (siehe Abschnitt 5.4.4).

[231] Das Design wurde auf Basis der beschriebenen Konzeption von einem professionellen Designer erstellt.

5.5 Implementierung des GHRC-Prototyps

Dieses Kapitel beschreibt die Implementierung der Stationensoftware und der weiteren Applikationen des GHRC-Prototyps gemäß der in Kapitel 5.4 beschriebenen Konzeption. In Abschnitt 5.4.2 wurde dargelegt, daß zur Umsetzung der identifizierten Anforderungen die Integration einer Workflow-Engine in die Stationensoftware benötigt wird.

5.5.1 Auswahl einer geeigneten Workflow-Beschreibungssprache

Die wesentlichen Anforderungen an eine geeignete Workflow-Beschreibungssprache wurden in Abschnitt 5.4.2 definiert. Im Zusammenhang mit der Implementierung des GHRC-Prototyps wurden Editoren verschiedener Workflow-Beschreibungssprachen und die zugehörigen Workflow-Engines evaluiert und einem Langzeittest unterzogen. Für den Langzeittest standen mehrere *virtuelle Maschinen* (*VM*s) auf einem leistungsstarken Virtualisierungsserver unter ‚VMWare ESX Server 3 Enterprise' zur Verfügung. Der Langzeittest sollte durch Abarbeitung einer Workflow-Definition durch die jeweilige Workflow-Engine eine einfache Beispielapplikation steuern. Zu diesem Zweck wurde ein Workflow mit einer Endlos-while-Schleife definiert, die eine if-else-Verzweigung enthält, deren Zweige in Abhängigkeit einer Zufallszahl gewählt werden. Jede ‚Start'-Aktivität ruft die Beispielapplikation auf und übergibt zwei Kommandozeilenparameter, die das Verhalten der Applikation steuern.

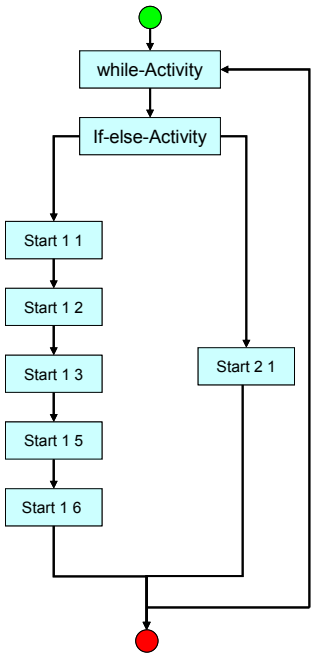

Abbildung 56: Illustration der Workflow-Definition für den Langzeittest von Workflow-Beschreibungssprachen und Workflow-Engines. Es handelt sich um einen sequenziellen Workflow in Endlosschleife. Die if-else-Bedingung wird in Abhängigkeit einer Zufallszahl ausgewertet. Jede der ‚Start'-Aktivitäten ruft eine Beispielapplikation auf, deren Verhalten durch zwei übergebene Kommandozeilenparameter gesteuert wird. Der Langzeittest erfolgte auf mehreren virtuellen Maschinen auf einem leistungsstarken Virtualisierungsserver unter ‚VMWare

ESX Server 3 Enterprise' und diente der Evaluierung der Stabilität von Workflow-Engines. In den folgenden Abschnitten werden die Ergebnisse zusammengefaßt.

5.5.1.1 Die *XML Process Definition Language* (*XPDL*)

Die *XML Process Definition Language* (*XPDL*)[232] ist eine XML-basierte Definitionssprache für die Repräsentation von Workflows. Ein Laborprotokoll ließe sich in der XPDL-Terminologie als sogenannter *XPDL-Prozeß*, einzelne Präparationspfade als *Subprocesses* oder *ActivitySets* definieren. Ein einzelner Präparationsschritt entspräche einer *Activity*. XPDL ist erweiterbar, erlaubt die Definition ereignisgesteuerter Transitionen zwischen einzelnen Aktivitäten, die Verwendung globaler Variablen hinsichtlich der Definition und Auswertung von Bedingungen zur Workflowsteuerung wie auch die Zuweisung von Aktivitäten zu Personengruppen, Applikationen oder konkreten Geräten unter Verwendung sogenannter *ToolAgents*. Zum Zeitpunkt der Evaluierung waren nur wenige kommerzielle und Open-Source XPDL-Workflow-Engines und -Editoren verfügbar[233]. Folgende Tabelle gibt eine zusammenfassende Übersicht der betrachteten XPDL-Editoren und –Engines:

Produkt	**Beurteilung**	**Langzeittest**
Editor : Enhydra JaWe (Java, Open Source, XPDL 1.0) Engine: Enhydra Shark (Java Open Source)	Editor: stabile Funktion, hinreichend dokumentiert, eingeschränkte Funktionalität, XPDL 1.0 Engine: hinreichende Dokumentation, Softwareinterfaces nach WfMC-Standard, eingeschränkte Funktionaliltät	Engine: bereits nach sehr kurzer Zeit Auftreten von Fehlern und Abbruch der Ausführung
Editor: Togethersoft Workflow Editor Demo (Java, kommerziell, XPDL 2.0) Engine: Togethersoft Workflow Server Demo (Java, kommerziell)	Editor: stabile Funktion, übersichtlich, gut und intuitiv bedienbar, hinreichend dokumentiert, Spezifikation XPDL 2.0 Engine: hinreichende Dokumentation, Softwareinterfaces nach WfMC-Standard, zusätzliche proprietäre Funktionen	Engine: Fehlermeldungen nach teilweise wenigen Minuten
Editor: Aspose Workflow Editor Demo (.net-Komponente, kommerziell, XPDL 1.0) Engine: Aspose Workflow Engine, Demo Version	Editor: schlechte Bedienbarkeit, kaum Dokumentation, Spezifikation XPDL 1.0 Engine: ungeeignet, da keine Dokumentation erhältlich	Engine: nicht getestet, da keine Dokumentation erhältlich

Tabelle 10: Zusammenfassung der evaluierten XPDL Workflow-Editoren und –Engines. Zum Zeitpunkt der Evaluierung waren nur wenige kommerzielle oder Open-Source-Implementierungen verfügbar. Die in der Tabelle aufgelisteten Produkte umfassen auch die von der WfMC erwähnten Implementierungen, konnten aber wegen instabiler Funktion oder wegen nicht verfügbarer Dokumentation nicht für die Implementierung des GHRC-Pro-

[232] XPDL wurde von der *Workflow Management Coalition* (*WfMC*) definiert, deren Arbeitsschwerpunkte unter anderem auf Geschäftsprozeßmanagement (siehe Abschnitt 2.2.4) liegen.

[233] Die auf der Website der WfMC erwähnten XPDL-Implementierungen wurden evaluiert.

totyps verwendet werden. Ein weiterer Grund für die Nicht-Verwendbarkeit war die Client/Server-Architektur der Implementierungen, die eine Integration einer Workflow-Engine in die Stationensoftware erschwert hätte.

Die fehlende Robustheit der getesteten XPDL-Workflow-Engines war besonders im Hinblick auf das Einsatzgebiet des GHRC-Prototyps und hinsichtlich des Projektzeitplans eines der Ausschlußkriterien. Ein anderes war die Client/Server-Architektur der evaluierten Workflow-Engines, die eine Integration der Workflow-Engine in die Stationensoftware erschwert hätte. Sowohl die Client- wie auch die Server-Komponente hätten jeweils in die Stationensoftware integriert werden müssen, um das in Abschnitt 5.4.2 beschriebene Konzept zu realisieren. Dies hätte zu einem gravierenden Overhead innerhalb der Stationensoftware geführt, denn die Workflow-Engines waren auf die Ablaufsteuerung komplexer Websites oder auf die Steuerung vollständiger Unternehmensabläufe ausgelegt, nicht aber auf die deutlich geringeren Anforderungen bei der Steuerung lediglich einzelner Aktivitäten auf jeweils einem Rechner. Aus den genannten Gründen wurde von den evaluierten Workflow-Engines und von XPDL als Workflow-Beschreibungssprache Abstand genommen.

5.5.1.2 Die *Windows Workflow Foundation* (*WWF*)

Als eine im Sinne von Abschnitt 5.4.2 geeignete Beschreibungssprache für die Repräsentation von Laborprotokollen hat sich die Workflow-Beschreibungssprache der *Windows Workflow Foundation* (*WWF*) herausgestellt. Die Windows Workflow Foundation unterstützt Workflow-Lösungen auf der Windows-Plattform und verwendet eine XML-basierte Beschreibungssprache zur Definition des Workflow-Modells. WWF erlaubt die Standardsprachen C# und *VisualBasic* zur Definition von *Activity-Code* eigener Aktivitäten. Die Workflow-Engine der Windows Workflow Foundation ist Teil des *Microsoft .Net Framework 3.0* und somit eine Komponente des Betriebssystems[234]. Eine wie in Abschnitt 5.4.2 konzipierte Einbettung der Workflow-Engine in die Stationensoftware, um autonome Laborstationen zu implementieren, war mit der WWF somit realisierbar. Ein geeigneter Workflow-Editor zum Erstellen von Workflow-Definitionen ist beispielsweise in *Visual Studio 2005* enthalten, kann aber auch in eigene Applikationen integriert werden. Die Windows Workflow Foundation unterstützt folgende Workflowtypen:

- Sequentielle Workflows
    - klar definierte Reihenfolge der Aktivitäten, parallele Aktivitäten möglich
      → geeignet zur Repräsentation von Laborprotokollen mit definiert aufeinander folgenden auszuführenden Präparationsschritten

[234] Somit sind keine zusätzlichen kommerziellen Produkte erforderlich. Diese Tatsache war für die Entscheidung jedoch nicht ausschlaggebend. Vielmehr wurde dadurch die Integration einer Workflow-Engine in die Stationensoftware erleichtert.

- Statusmechanismus-Workflows:
    - entsprechen einer Zustandsmaschine
    - Reihenfolge der Aktivitäten bei der Abarbeitung ist Zustands- und Ereignis-gesteuert, bestehen aus einer Vielzahl von:
        - Zuständen
        - regelbasierten Transitionen

Auch die Windows Workflow Foundation wurde dem in Abschnitt 5.5.1 beschriebenen Langzeittest unterzogen. Die erfolgreiche Ausführung jedes Zyklus wurde in einer .log-Datei mit Datum, Uhrzeit und Zufallswertwert protokolliert. Dieser Test lief einige Monate lang stabil und fehlerfrei. Er wurde schließlich nach mehreren Hunderttausend Zyklen beendet.

5.5.1.3 Dezentrale Repräsentation vollständiger Präparationsprotokolle

Grundsätzlich besteht eine Workflow-Definition in WWF aus zwei Komponenten:

- dem Workflow-Modell (Beschreibung der Aktivitätenabfolge)
- dem Klassencode (Workflow-Code bzw. Activity-Code)

Für die Ausführung eines Workflows ist die Vorkompilierung des Klassencodes in eine sogenannte *.net-Assembly* erforderlich, der dann in der sogenannten *Common Intermediate Language* (*CIL*) vorliegt[235]. Für die Repräsentation von Workflow-Definitionen bietet die Windows Workflow Foundation mehrere Möglichkeiten, denn das Workflow-Modell kann grundsätzlich auf zwei verschiedene Arten repräsentiert werden, nämlich:

- als Code (C#, VB.net)
- als proprietäres *XML-Workflow-Markup* (*'XOML'*)

Wird das Workflow-Modell als Code repräsentiert, dann wird es zusammen mit dem Klassencode in eine gemeinsame .net Assembly vorkompiliert. Diese Assembly stellt dann die vollständige Workflow-Definition dar. Wird das Workflow-Modell jedoch als XML-Workflow-Markup repräsentiert, dann gibt es verschiedene Varianten, das Markup mit dem Klassencode zu kombinieren. Eine Variante besteht darin, auch das Markup gemeinsam mit dem Klassencode in eine .net Assembly vorzukompilieren. Im Zusammenhang mit der beabsichtigten dezentralen Speicherung von Workflow-Definitionen auf den Speicherchips der Probenröhrchen (siehe Kapitel 4.5) oder auf den gemeinsamen Speicherchips von Trägerplatten oder Lagerboxen (siehe Abschnitt 5.4.3) ist jedoch diejenige Variante interessant, bei der

---

[235] Beispielsweise in Form einer ausführbaren Datei (.exe) oder einer dynamic link library (.dll). CIL ist die native Sprache des .net-Frameworks für schrittweise Kompilierung durch *Just-in-time*-Compiler unmittelbar vor der Ausführung durch die *Common Language Runtime* (*CRT*).

ausschließlich der Klassencode vorkompiliert und das Workflow-Modell in Form des XML-Markups beibehalten wird, also nicht vorkompiliert wird. In diesem Fall spricht man von *Codetrennung*. Codetrennung bezeichnet hier eine Trennung von Workflow-Modell und vorkompiliertem Klassencode mit der Konsequenz, daß das Workflow-Markup getrennt vom (vorkompilierten) Klassencode gespeichert werden kann. Der vorkompilierte Klassencode kann dann beispielsweise in Form einer .dll in die Stationensoftware integriert werden, während das Workflow-Modell als XML-Markup auf dezentralen Speicherchips abgelegt werden kann. Die Optionen des Workflow-Modells sind in folgender Abbildung dargestellt:

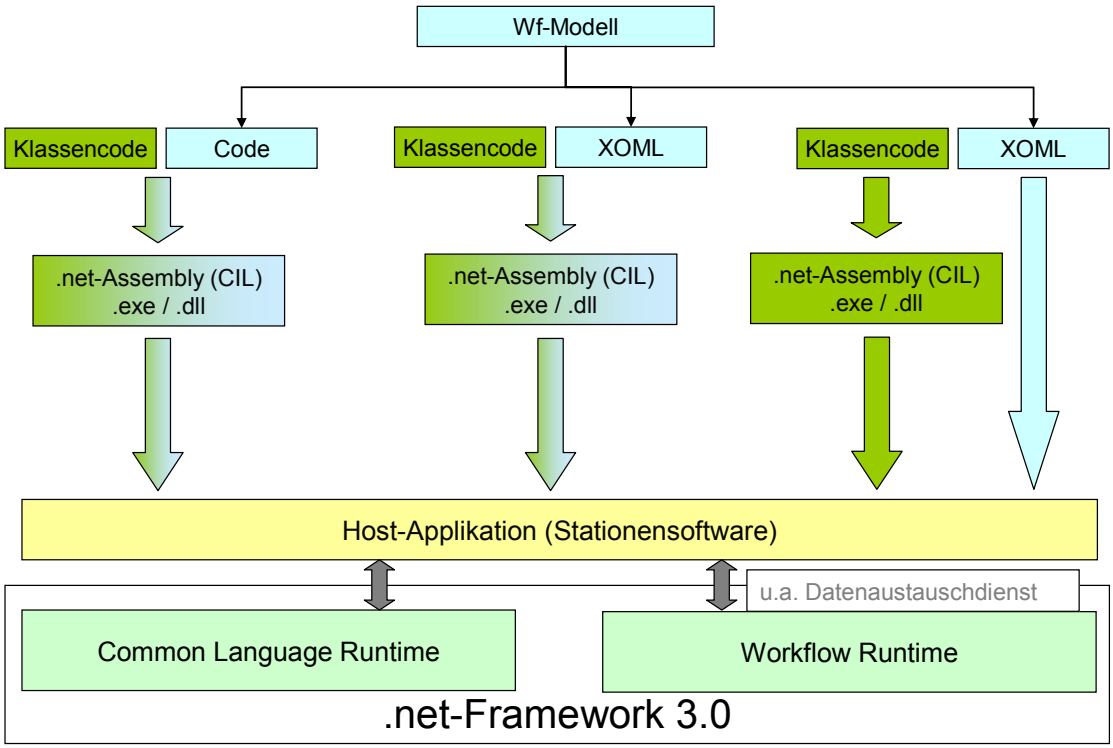

Abbildung 57: Optionen des Workflow-Modells. Die Abbildung illustriert die verschiedenen Möglichkeiten, Workflow-Definitionen in der Windows Workflow Foundation zu repräsentieren. In WWF besteht jede Workflow-Definition aus den beiden Komponenten Klassencode und Workflow-Modell. Aus den verschiedenen Möglichkeiten, das Workflow-Modell zu repräsentieren, ergeben sich drei Varianten: (Links): wird das Workflow-Modell als Code repräsentiert, dann wird es zusammen mit dem Klassencode in eine gemeinsame .net Assembly vorkompiliert in Common Intermediate Language vorkompiliert. Diese Assembly stellt dann die vollständige Workflow-Definition dar, die von einer Host-Applikation in Verbindung mit der Workflow Runtime der WWF und der Common Language Runtime des Betriebssystems ausgeführt wird (Mitte): Wird das Workflow-Modell als XML-Workflow-Markup repräsentiert, kann es ebenfalls gemeinsam mit dem Klassencode in eine gemeinsame .net Assembly vorkompiliert werden. (Rechts): Wird das Workflow-Modell als XML-Workflow-Markup repräsentiert, kann es aber auch in Form des XML-Markups beibehalten werden und wird nicht vorkompiliert (Codetrennung). In diesem Fall wird nur der Klassencode vorkompiliert und kann in die Stationensoftware integriert werden mit der Konsequenz, dass das Workflow-Modell (räumlich) getrennt vom vorkompilierten Klassencode gespeichert werden kann, beispielsweise auf seriellen Flash-Speicherchips. In dieser Arbeit bedeutsam ist die Variante mit Codetrennung, da sie auf dezentralen Speicherchips lediglich Speicherplatz für das XML-Workflow-Markup benötigt.

Um die Variante mit Codetrennung zu testen, wurde eine einfache Host-Applikation implementiert, die das Laden und Instanziieren eines XML-Workflow-Markups erlaubt. Eine eigenentwickelte, rudimentäre Aktivität für die Interaktion mit Personen wurde definiert und ihr vorkompilierter Klassencode (Activity-Code) als .net-Assembly in die Host-Applikation integriert. Damit wurde der Funktionsumfang der Host-Applikation um die Funktionalität des Activity-Codes erweitert. Ein Aufruf dieser Funktionalität und eine Parameterübergabe erfolgen durch Abarbeitung der entsprechenden Aktivität im XML-Workflow-Markup durch die Workflow-Runtime und durch deren Kommunikation[236] mit der Host-Applikation. Der vorkompilierte Klassencode der Aktivität wird durch die Common Language Runtime ausgeführt. Im Zusammenhang mit dem Test auf Machbarkeit werden durch die eigenentwickelte Aktivität eine parametrierte Ausgabe an den Benutzer und ein Ereignishandling (Reaktion auf das Betätigen von Buttons) implementiert. Die Ereignisse werden als Parameter von der Hostapplikation an die Workflow-Instanz übergeben, so daß die Workflow-Instanz durch Interaktion von Host-Applikation und Benutzer wiederum parametriert wird. Der Ansatz, eine Host-Applikation modular um benötigten Activity-Code zu erweitern, die Abfolge der Aktivitäten durch XML-Workflow-Markups zu beschreiben und Activity-Code durch die Abarbeitung von Workflow-Markups[237] auszuführen, wurde damit erfolgreich getestet. Damit ist bei der Windows Workflow Foundation die in Kapitel 5.4 geforderte Erweiterbarkeit einer Beschreibungssprache um eigene Aktivitäten wie auch die Möglichkeit der vollständig dezentralen Speicherung von Workflow-Definitionen gegeben.

Die Windows Workflow Foundation verfügt über Funktionen für die automatische Statusprotokollierung der Abarbeitung einer Workflow-Instanz. Diese Funktionen erlauben auch das Speichern des Status einer Workflow-Instanz in einer Datei. Dies ermöglicht einerseits die Wiederaufnahme der Bearbeitung einer Workflow-Instanz an genau derjenigen Stelle, an der sie zuvor unterbrochen wurde, und andererseits den Austausch des Bearbeitungsstatus mit anderen Instanzen der Workflow-Engine. In Bezug auf das Konzept der autonomen Laborstationen sind dies exakt die Mechanismen, die benötigt werden, um eine Präparation an einer Laborstation zu unterbrechen[238] und an einer anderen Laborstation wiederaufzunehmen. Mit Hilfe der Speicherchips von Cryo-Devices kann der Status der Abarbeitung der Workflow-Instanz zwischen den Laborstationen transportiert werden, wie in der Konzeption des GHRC-Prototyps in Abschnitt 5.4.3 beschrieben.

---

[236] Über eine Schnittstelle (*Datenaustauschdienst*)

[237] durch eine Workflow-Engine

[238] siehe auch Abschnitt 5.4.3.

5.5.1.4 Eigene Aktivitäten und der Workflow-Editor

Auf Basis der oben evaluierten Eignung der Windows Workflow Foundation gemäß den in Abschnitt 5.4.2 formulierten Kriterien wurde Activity-Code unter anderem für die folgenden eigenen Aktivitäten[239] implementiert:

- HumanActivity: Die HumanActivity dient der Ausgabe von Handlungsanweisungen an den Anwender. Jeder Protokollschritt, der von Anwendern auszuführen ist, wird im XML-Workflow-Markup als HumanActivity definiert. Parameter (unter anderem):
  - LongMessage: Zeichenkette, die Benutzerinstruktionen und HTML-Tags zur formatierten Ausgabe der Instruktionen enthält
  - ShortMessage: Zeichenkette, die der Beschriftung der GUI-Registerkarte der Aktivität dient. Dies erleichtert die Orientierung und das Blättern zwischen Aktivitäten (siehe Abschnitt 5.4.6.3).
  - AllowedActions: einige boolesche Parameter, die ausgewählte Bedienelemente des GUI bereits in der Workflow-Definition ein- oder ausschalten. Dies ermöglicht eine Protokollschritt-angepaßte Darstellung der Benutzeroberfläche gemäß der GUI-Konzeption (siehe Abschnitt 5.4.6.3), um dem Benutzer nur die potentiell benötigten GUI-Elemente zu präsentieren, beispielsweise
    - RegisterReagent: Button zum Erfassen von Reagenzdaten
    - WorkOnSamples: blendet mehrere Bedienelemente (Done, Lost, SetCount, PrintLabel) ein oder aus, die unmittelbar mit dem Proben- und Aliquotenmanagement verbunden sind und Einfluß auf die verschiedenen ID-Listen haben bzw. das bedarfsorientierte Replizieren von Etiketten erlauben
- HumanMultiInputActivity: dient der Definition des zu akquirierenden Proben- und Prozeßwissens bereits in der Workflow-Definition in Form von Eingabefeldern, und zudem der Übergabe dieser Werte als globale Variablen an die Workflow-Instanz[240].

---

[239] Bei den hier vorgestellten Aktivitäten handelt es sich um diejenigen, die den Funktionen des ‚Generic Device' für die Interaktion mit dem Anwender entsprechen. Im Unterschied zu den assoziierten HTML-Dateien und den vordefinierten Formularen im Testsystem erlauben sie eine vollständig dezentrale Definition von Instruktionen und Eingabefeldern im XML-Workflow-Markup. Die Ausführung der jeweiligen Aktivität im Workflow-Markup erfolgt durch Aufruf des zugehörigen, in die Stationensoftware integrierten Activity-Codes durch die Workflow-Runtime.

[240] Auf Basis dieser Variablen können beispielsweise Bedingungen durch die Workflow-Engine überprüft und der weitere Verlauf der Ausführung beeinflußt werden.

Der Aufruf des Klassencodes der HumanMultiInputActivity hinterlegt die Namen und Datentypen der zu erfassenden Werte im Dokumentationseditor der Stationensoftware. Der Editor kann mit Hilfe einer GUI-Schaltfläche beliebig ein- und ausgeblendet werden, zeigt die Datenfelder an, erlaubt die Werteeingabe, die automatische Speicherung und nachträgliche Änderungen. Die Werte werden in einem speziellen Elementtyp innerhalb des Workflow-spezifischen Target-Files gespeichert. Parameter (unter anderem):

- Texts: String-Array zur Definition der Namen der zu akquirierenden Werte. Die Anzahl der enthaltenen Strings ist variabel.
- Types: Typ-Array zur Definition der gültigen Datentypen für die zu akquirierenden Werte, die durch ‚Texts' benannt worden sind.
- InitialValues: String-Array[241] zur optionalen Vorbelegung der einzelnen Eingabefelder.

- Finalize-Activity: unmittelbar nach der Aliquotierung einer Probe in diskrete Substrate oder Standardröhrchen mit ‚Portlink' werden die Workflow-Definition, der Workflow-Status und das aktuelle Target-File auf die integrierten seriellen Flash-Speicherchips geschrieben. Für diesen Vorgang wurde die Bezeichnung *Finalisierung* gewählt. Die Finalize-Activity umfaßt mehrere Vorgänge:
  - Registrieren der Seriennummer von ‚Icebreaker' oder ‚Portlink'
  - Erfassen der ProbenID der enthaltenen Probe
  - Abfragen expliziten Probenwissens aus der ‚EurocryoDB' per Web-Service
  - Beschreiben diskreter Substrate

  Bei Abarbeitung einer Finalize-Aktivität im Workflow-Markup durch die Workflow-Runtime wird der entsprechende Activity-Code ausgeführt: der aktuelle Status der Workflow-Instanz, die akquirierte Dokumentation in Form des aktuellen Target-Files und per Web-Service aus der Probendatenbank abgefragte Probendaten werden aus dem Cache[242] der Laborstationen auf die seriellen Speicherchips von ‚Icebreakern' und ‚Portlinks' geschrieben. Dazu interagiert die Finalize-Activity mit Hilfe des in Abschnitt 5.4.4 konzipierten Device Integration Frameworks *DCMS* mit den in Abschnitt

[241] Es handelt sich hier um drei getrennte String-Arrays, da der Datenaustausch mit der Workflow-Engine nur durch einfache Datentypen erfolgen kann.

[242] Die Stationensoftware nutzt einen lokalen Cache zur Zwischenspeicherung der Workflow-Definition, des Status und des Target-Files.

5.4.3 gezeigten Dockingstationen[243]. Die Finalize-Activity wird in einem GUI-Plugin[244] dargestellt.

Der Activity-Code der beschriebenen eigenen Aktivitäten wurde in Form einer zusätzlichen *Activity Library* organisiert. Diese ergänzt die Activity Library der Windows Workflow Foundation[245]. Ihr Klassencode wurde in Form einer .net-Assembly (siehe Abschnitt 5.5.1.3) vorkompiliert und kann gemäß dem oben beschriebenen Prinzip der Codetrennung von Aktivitäten in Workflow-Definitionen genutzt werden.

Um die Anforderungen an eine Applikation zum Erstellen von Workflow-Definitionen (siehe Abschnitt 5.4.5) zu erfüllen, wurde ein eigener Editor implementiert, der

- die Verwendung von Standard-Aktivitäten und eigenen Aktivitäten zum Erstellen von Workflow-Definitionen (in Form von XML-Markups[246]) ermöglicht[247]

- zusätzlich zur Workflow-Definition eine Übersichtsgrafik, eine Datei mit aussagekräftiger Workflow-Beschreibung und ein Target-File-Template erzeugt, das sich am Workflow-Markup orientiert (siehe Abschnitt 5.4.4)

- XML-Workflow-Markup, Übersichtsgrafik, Beschreibungsdatei und Target-File-Template in einer gemeinsamen Container-Datei (‚Workflow.zip') organisiert. Die Container-Dateien der verschiedenen Protokolle werden in einem gemeinsamen Verzeichnis abgelegt[248].

Die Struktur eines Target-Files spiegelt die Aktivitäten-Abfolge innerhalb der Workflow-Definition wider. Während das XML-Workflow-Markup alle Parameter (beispielsweise Instruktionen an den Benutzer, Meta-Geräteparameter, GUI-Parameter) enthält, besteht das Target-File aus Informationen über die erwartete Protokoll-Abarbeitung und die erwarteten

---

[243] Die Ansteuerung der Dockingstationen per DCMS wird im Detail in Abschnitt 5.5.2.2 beschrieben.

[244] gemäß Konzeption in Abschnitt 5.4.4.

[245] Durch Ergänzen weiteren Activity-Codes in der Activity Library kann die Menge der zur Verfügung stehenden Aktivitäten zum Erstellen von Workflow-Definitionen jederzeit nach Bedarf erweitert werden. Activity-Code kann auch in weiteren zusätzlichen Activity Libraries organisiert werden.

[246] Das XML-Workflow-Markup der WWF besteht aus zwei Dateien: (1) einer Datei (‚XOML') zum Beschreiben der Aktivitäten und ihrer Abfolge und (2) einer Datei (‚rules') zum Definieren von Bedingungen und Transitionen auf Basis globaler Variablen oder GUI-Events.

[247] Der Editor greift zum Erstellen des XML-Workflow-Markups auf die Activity Libraries zu.

[248] Auf dieses Verzeichnis greift auch die in Abschnitt 5.4.5 konzipierte Initialisierungsapplikation zu. Sie schreibt die vollständige Containerdatei auf die Cryo-Devices und instanziiert das darin enthaltene Target-File mit der initialen ToDo-Liste.

Ergebnisse der im Workflow-Markup enthaltenen Aktivitäten. Für jede Aktivität in der Workflow-Definition existiert ein Datenblock im Target-File. Unterschiedliche Aktivitäten können unterschiedliche Target-Strukturen besitzen. Das jeweilige Target-File-Template basiert auf vordefinierten Target-Strukturen für die jeweiligen Aktivitäten. Die in Target-Files verwendeten Elemente sind im Wesentlichen:

- Workflow.WorkflowID: identifiziert das ausgeführte Protokoll
- Workflow.Version: Version der Workflow-Definition
- ListTargetSteps: Liste der im Protokoll enthaltenen Bearbeitungsschritte
    - TargetStep.ActivityID: identifiziert einen Eintrag im Target-File anhand der Aktivität der Workflow-Definition
    - TargetStep.ActivityType: gibt an, um welchen Typ von Aktivität es sich bei dem Bearbeitungsschritt handelt.
    - TargetStep.Samples: eine Liste von SampleItems
    - SampleItem: Container für Probeninformation
        - SampleItem.SampleID: ist eine einzelne ProbenID (wenn die Proben zu diskreten Zeitpunkten der Aktivität unterzogen wurden) oder ist eine Liste von ProbenIDs oder AliquotIDs (wenn alle Proben gleichzeitig der Aktivität unterzogen wurden, beispielsweise beim Zentrifugieren). Wenn mehrere Proben der Aktivität nacheinander unterzogen werden, wird für jede Probe ihr Bearbeitungszeitpunkt aufgelistet.
        - SampleItem.Status: Status der Präparation der Probe, mögliche Werte sind: Done, Lost, New
    - TargetStep.StepStatus: Status der Aktivität. Mögliche Werte sind: Planned, In progress, Completed, Error
    - TargetStep.TimeStampBegin und TimeStampEnd: Zeitpunkte, zu denen die Aktivität begonnen bzw. abgeschlossen wurde
    - TargetStep.Comment: Text zum Dokumentieren eventueller Abweichungen vom vorgegebenen Präparationsschritt

5.5.2 Ein Device Integration Framework für die Laborstationen: ‚DCMS'

Aufbauend auf dem in Abschnitt 5.5.1.3 geschilderten Konzept der Codetrennung von Workflow-Modell und Activity-Code und basierend auf dem favorisierten Ansatz zur dezentralen Speicherung des Workflow-Markups und der Integration vorkompilierten Activity-Codes in die Hostapplikation durch zusätzliche Activity Libraries ist unter Verwendung der Windows Workflow Foundation der in Abschnitt 5.5.1.4 beschriebene Edtitor zum Erstellen von Workflow-Definitionen entstanden. Er erzeugt eine Containerdatei, deren Bestandteile (Workflow-Definition, Workflow-Beschreibung, Target-File) die Voraussetzung für die Umsetzung des Konzepts autonomer Laborstationen sind. Nach ihrer Erstellung durch den Editor liegen die verschiedenen Containerdateien in einem gemeinsamen Repository vor[249].

Der Ausführung einer Workflow-Instanz durch eine Instanz der Stationensoftware gehen zwei Punkte voraus:

- Die entsprechende Containerdatei muß auf die gemeinsamen Speicherchips derjenigen Trägerplatte geschrieben werden, die zur Organisation der zu bearbeitenden Proben verwendet wird.[250]). Das Target-File wird instanziiert, d.h. um die initiale ToDo-Liste ergänzt. Diese Vorgänge erfolgen durch die Initialisierungsapplikation (siehe Abschnitt 5.4.5).

- Um die Trägerplatte überhaupt mit der Initialisierungsapplikation oder der Stationensoftware nutzen zu können, muß sie zuvor formatiert werden. Dies erfolgt durch die Formatierungsapplikation (siehe auch Abschnitt 5.4.5).

Die Interaktion mit Trägerplatten, Lagerboxen, ‚Icebreakern' oder ‚Portlinks' ist somit nicht auf die Stationensoftware beschränkt. Formatierungsapplikation und Initialisierungsapplikation interagieren ebenfalls mit Cryo-Devices und erfordern die Integration der in Abschnitt 5.4.3 konzipierten Dockingstationen. Sinngemäßes gilt für die Interaktion mit Barcodescannern und Barcodedruckern zur effizienten Identifizierung und Kennzeichnung von Cryo-Devices. Es liegt daher nahe, wiederverwendbare, modulare Softwarekomponenten auf Gerätelevel zur Integration dieser Geräte zu verwenden. In Abschnitt 5.4.4 wurde hinsichtlich unterschiedlicher Geräteausstattung von Laborstationen zusätzlich die modulare und flexible Anbindung von Geräten durch Meta-Applikationen, die mittels Metabefehlen aus Workflow-Aktivitäten heraus gesteuert werden sollen, konzipiert. Für die Interaktion von Geräten mit den Systemkomponenten des GHRC-Prototyps wurde daher eine entsprechende Infrastruktur[251] zur Geräteintegration implementiert, die das Senden von Meta-Kommandos an Geräte-

---

[249] Im Fall des GHRC-Prototyps ist dies ein Verzeichnis.

[250] Siehe auch Abschnitt 5.4.3.

[251] Es handelt sich um ein Device Integration Framework im Sinne von Abschnitt 3.4.2.2.

applikationen sowohl aus GUI-Applikationen wie auch aus Workflow-Aktivitäten heraus erlaubt.

Die benötigte Laborunabhängigkeit von Workflow-Definitionen erfordert Kommandos und Parameter, die von allen Geräten der jeweiligen Geräteklasse unterstützt werden. Gleiches gilt für GUI-Applikationen, aus denen heraus Geräte angesteuert werden sollen. Aus Sicht der Workflow-Engine (oder einer GUI-Applikation) ist nur relevant, daß ein Kommando in benötigter Weise von einem Gerät der entsprechenden Geräteklasse ausgeführt werden kann. Die Verteilung von Kommandos und deren korrekte Ausführung kann somit durch Software-Komponenten erfolgen, die zwischen Applikationslevel und konkretem Laborgerät agieren. Diese Überlegungen waren Ausgangspunkt für die Implementierung des nachfolgend beschriebenen *Device Control Management Systems* (*DCMS*). Das DCMS ist ein hierarchisch strukturiertes Device Integration Framework (siehe auch Abschnitt 3.4.2.2), das die nachfolgend beschriebenen Hierarchieebenen umfasst:

- Als *Commander* wird diejenige Hierarchieebene bezeichnet, von der die auszuführenden Meta-Kommandos mit ihren Parametern an Geräte ausgehen. Dies kann die Workflow-Engine der Stationensoftware sein (durch Abarbeitung Geräteklassen-bezogener Meta-Aktivitäten, siehe Abschnitt 5.4.4) oder GUI-Applikationen wie beispielsweise Initialisierungs- oder Formatierungsapplikation. Der Commander stellt damit innerhalb des DCMS die Wurzel dar und ist entweder eine GUI-Applikation oder eine Workflow-Engine.
- *DCManager* dienen der Steuerung gerätespezifischer Applikationen (*DCUnits*, siehe unten) und der Kommunikation zwischen DCUnits und dem Commander. DCManager verfügen über eine Liste der ihnen zugeordneten DCUnits, die dynamisch über deren Betriebszustände informiert. Anhand dieser Liste kann ein DCManager den Befehlssatz (dieser entspricht der Menge der implementierten *DCJobs* einer DCUnit, siehe unten) der jeweiligen DCUnits abfragen[252] und eine Liste ihrer Funktionalitäten erstellen. Dies ist für die weiter unten beschriebene Interaktion der Hierarchieebenen erforderlich.
- *Device Control Units* (*DCUnits*) sind die ausführenden Einheiten in der Hierarchie. Es handelt sich bei ihnen um gerätespezifische Applikationen, die einerseits mit Hilfe ihres übergeordneten DCManagers mit dem Commander kommunizieren und andererseits die konkrete Ansteuerung eines spezifischen Geräts durchführen, entweder direkt oder mittels eines Gerätetreibers. Der jeweilige Befehlssatz und die Befehlssyntax des entsprechenden Laborgerätes sind in der zugehörigen DCUnit implementiert. Für jedes Gerät wird eine spezifische Instanz der Klasse DCUnit implementiert, die auch ein

[252] Dies setzt voraus, daß die DCUnits ihre implementierten DCJobs kennen und dem DCManager mitteilen können.

eigenes Service-GUI für die unmittelbare Bedienung des Geräts unabhängig von Commander und DCManager zur Verfügung stellt.

- Die konkrete Ansteuerung von Geräten durch DCUnits ist aufgabenbasiert und erfolgt durch gekapselte, entsprechend sinnvolle Kombinationen einzelner Elemente des implementierten gerätespezifischen Befehlssatzes. Eine solche Kombination wird als *DCJob*[253] bezeichnet, und ihre Ausführung erfüllt eine bestimmte Aufgabe, beispielsweise das Lesen des Speicherchips eines ‚Icebreakers' auf einer Trägerplatte mit Hilfe einer Dockingstation. Dieser beispielhaft erwähnte DCJob ‚ReadFlash' steuert das Verhalten der Dockingstation und setzt sich aus mehreren einzelnen Befehlen zusammen (‚Power on', ‚Select Chip', ‚Select MemoryPage', ‚Read'), die Teile des implementierten Geräte-Befehlssatzes sind, und die in anderer Zusammenstellung für andere Aufgaben verwendet werden können. Der Kommandosatz einer DCUnit ist die Menge der von ihr ausführbaren Aufgaben, also die Menge der implementierten DCJobs, und definiert somit, welche Aufgaben die DCUnit und damit das zugeordnete Laborgerät ausführen kann. DCJobs sind als selbstorganisierte Module implementiert, die Ergebnisse, Fortschritt und Fehler an die DCUnit berichten.

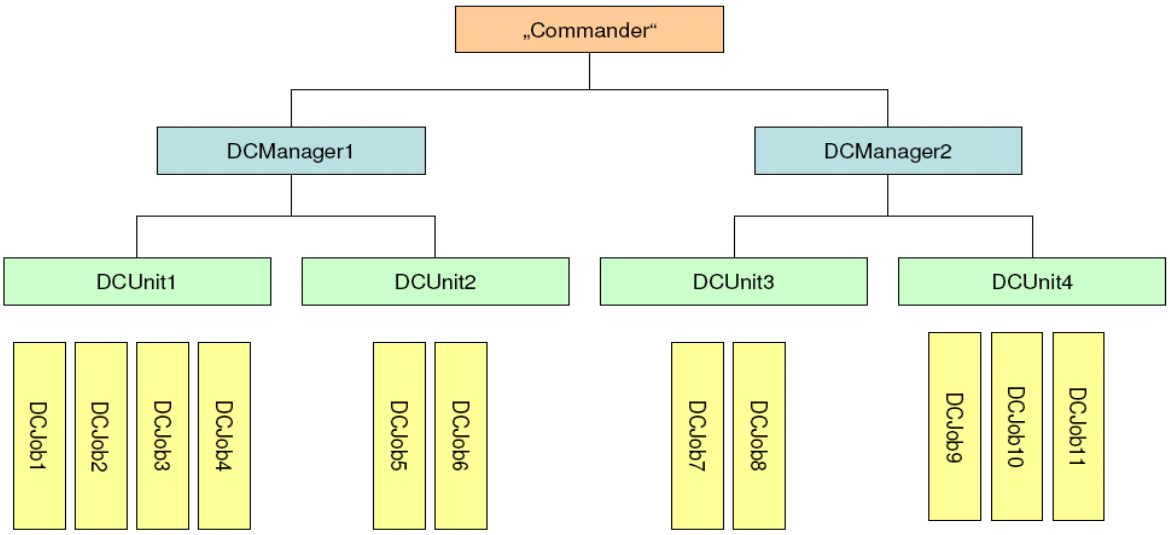

Abbildung 58: Hierarchischer Aufbau des DCMS aus Commander, mehreren DCManagern, jeweils zugehörigen DCUnits und deren gerätespezifischen Befehlssätzen in Form von DCJobs. Als Commander können beispielsweise Workflow-Engines oder GUI-Applikationen dienen. Von ihm gehen die Meta-Kommandos mit Meta-Parametern an Geräte aus. DCManager steuern die gerätspezifischen DCUnits und dienen der Kommunikation zwischen Commander und DCUnits. Anhand einer Liste der ihnen zugeordneten DCUnits können DCManager die Funktionalitäten der DCUnits und ihre Befehls- und Parametersätze abfragen. DCUnits sind gerätespezifische Applikationen, die mit Hilfe ihres übergeordneten DCManagers mit dem Commander kommunizieren und die konkrete Ansteuerung eines spezifischen Geräts durchführen. Dieser Ansteuerung dienen gekapselte DCJobs, die aus Elementen des in der jeweiligen DCUnit implementierten gerätespezifischen Befehlssatzes bestehen.

[253] Die gesamte Funktionalität zur Ansteuerung eines konkreten Laborgeräts wird in Form geeigneter DCJobs implementiert und ist Teil der gerätespezifischen Instanz von DCUnit.

5.5.2.1 Interaktion der Hierarchie-Ebenen

Jedes Gerätekommando geht vom oben beschriebenen Commander aus und kann seinen Ursprung entweder in der Abarbeitung einer Meta-Aktivität einer Workflow-Definition oder in einem GUI-Event haben. Der Commander sendet eine Anfrage an den DCManager, ob das auszuführende Gerätekommando einem der implementierten DCJobs einer seiner verwalteten DCUnits entspricht. Anhand der im DCManager vorhandenen Liste der DCUnits und ihrem jeweiligen Kommandosatz gibt der DCManager gegebenenfalls den Namen einer kompatiblen (d.h. das Kommando beherrschenden) DCUnit an den Commander zurück. Der Commander fordert beim DCManager dann ein *JobTemplate* dieser DCUnit für genau dieses Kommando an. Das JobTemplate definiert die Syntax (Kommando und Struktur der Eingabeparameter) für genau diesen DCJob genau dieser DCUnit. Metakommando und Metaparameter werden vom Commander in das JobTemplate eingetragen[254]. Das parametrierte JobTemplate wird vom Commander an den DCManager zurückgesendet und dessen Ausführungsmethode[255] wird aufgerufen. Der DCManager sendet den DCJob an eine Warteschlange der kompatiblen DCUnit, die anstehende DCJobs gemäß ihrer Regularien (die Regularien des verwendeten Laborgerätes widerspiegeln) abarbeitet. Die Abarbeitung von DCJobs erfolgt in jeweils eigenen Threads[256] und ermöglicht somit ein Multithreading der DCUnit (siehe auch Abschnitt 3.4.1.1). Nach erfolgter Abarbeitung meldet der DCJob seinen Status an seine übergeordnete DCUnit, die den DCManager über den erledigten DCJob informiert. Der DCManager informiert den Commander, der die Ergebnisse des DCJobs abruft[257].

Als eine der Anforderungen an den GHRC-Prototypen war Robustheit definiert worden (siehe Abschnitt 5.4.4). Der Ausfall eines Gerätes oder seiner DCUnit darf die Funktion der Stationensoftware oder die Funktion anderer Laborgeräte nicht beeinträchtigen. Ebenso waren Flexibilität hinsichtlich der modularen Geräteausstattung und Wiederverwendbarkeit von Softwarekomponenten als weitere Anforderungen an ein Device Integration Framework identifiziert worden. Diese Robustheit, Modularität und Erweiterbarkeit können dadurch erreicht werden, daß Commander und DCUnits in jeweils verschiedenen Prozessen ausgeführt wer-

---

[254] Dadurch wird ein für eine Geräteklasse allgemein gültiges Meta-Kommando mit seinen Parametern in einen konkret von einer DCUnit ausführbaren DCJob transformiert. Dies erlaubt eine von konkreten Laborgeräten entkoppelte Repräsentation des Kommandos und seiner Parameter als Aktivität im XML-Workflow-Markup.

[255] Der DCManager kennt grundsätzlich zwei verschiedene Ausführungsmethoden: die asynchrone DoJob() und die synchrone DoJobB()-Methode, entsprechend der gewünschten Ausführungsform des DCJobs. Man unterscheidet synchrone (blockierende) und asynchrone (nicht blockierende) Ausführung von DCJobs. Eine synchrone Ausführung blockiert die Ausführung anderer DCJobs an derselben DCUnit solange, bis er abgearbeitet ist. Ein asynchroner DCJob erlaubt weitere DCJobs an derselben DCUnit.

[256] Das Thread wird durch die Methode Run() des jeweiligen DCJobs kreiert.

[257] Dies erfolgt mit dem Befehl GetResults().

den. Der Informationsaustausch zwischen Commander und DCUnits erfolgt mit Hilfe zusätzlicher Kommunikationsschichten (zwischen Commander und DCManager: *Client Connector*, zwischen DCManager und DCUnits: *Server Connector*) über TCP/IP-Socketverbindungen. Der DCManager selbst ist zweigeteilt und besteht aus einem *DCManagerClient* (in der Stationensoftware integriert) und einem *DCManagerServer* pro DCUnit (in der jeweiligen DCUnit integriert)[258].

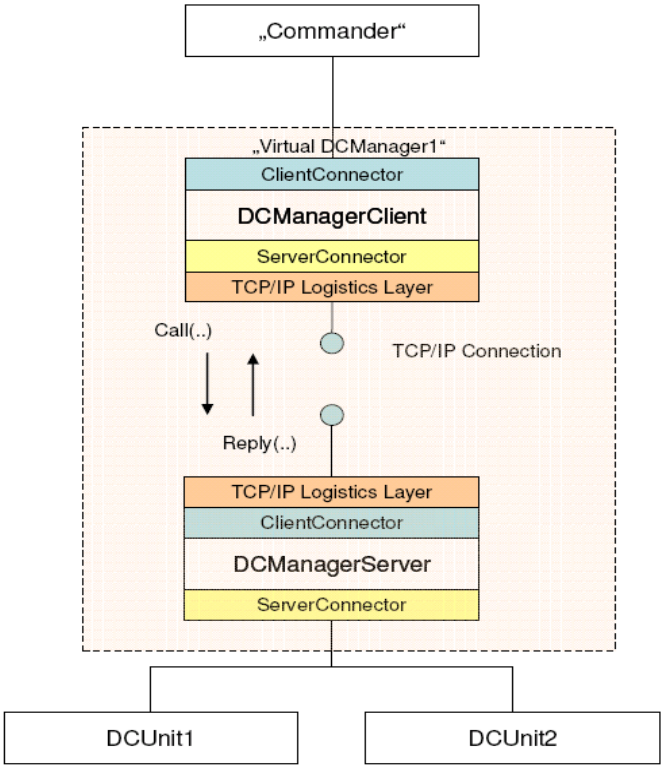

Abbildung 59: zweigeteilter DCManager. Der Commander (beispielsweise die Workflow-Engine) kommuniziert mit dem DCManagerClient der Stationensoftware. Dieser kommuniziert über TCP/IP-Sockets mit dem DCManagerServer, der in der jeweiligen DCUnit integriert ist. Die DCUnit führt DCJobs auf dem konkreten Gerät aus. Eine exemplarische Aufrufsequenz für einen DCJob: (1) Commander ruft DCManager1.DoJob(DCJob) auf. (2) DCManagerClient ruft ServerConnector.Call(…) auf. (3) Die Implementierung des ServerConnectors (hier: der TCP/IP Support-Layer) sendet via Socket-Verbindung den (serialisierten) DCJob zum empfangenden TCP/IP Support-Layer des Server-Prozesses (DCUnit). (4) Der TCP/IP Support-Layer des Server-Prozesses deserialisiert den empfangenen Bytestream in einen DCJob und ruft ClientConnector.Call(DCJob) auf. (5) ClientConnector.Call(DCJob) ruft DCManagerServer.DoJob(DCJob) auf. (6) Der Job wird von der DCUnit ausgeführt. (7) DCManagerServer beendet die Ausführung des Jobs und ruft ClientConnector.Reply(DCJob) auf. (8) Der DCJob wird serialisiert und an den den TCP/IP Support-Layer des DCManagerClient zurückgesendet. (9) Der TCP/IP Support-Layer des DCManagerClient empfängt den Bytestream und deserialisiert ihn in einen DCJob. (10) DCManagerClient.Reply(DCJob) wird aufgerufen.

[258] Die Aufspaltung des DCManagers in Client-seitigen Teil und Server-seitigen Teil und die Kommunikation der beiden Teile über Socket-Verbindungen machen das DCMS grundsätzlich netzwerkfähig, so daß Commander und DCUnits auch auf unterschiedlichen Rechnern im Netzwerk ausgeführt werden können.

Das beschriebene Device Integration Framework ist flexibel, erweiterbar und modular. Es basiert auf definierten Kommandosätzen mit jeweiligem Parametersatz (die jeweiligen DCJobs der verschiedenen DCUnits), so daß eine definierte Schnittstelle gegeben ist, die von Applikationen oder Workflow-Engines verwendet werden kann. Dies erlaubt den in Kapitel 5.4 konzipierten Systemkomponenten, die Rolle des Commanders auszuüben. Das Anfordern der Parameterstruktur von DCJobs kompatibler DCUnits durch den Commander und die anschließende Transformation von Meta-Kommandos und Meta-Parametern ermöglichen die Verwendung von Meta-Aktivitäten in Workflow-Definitionen. Dies erlaubt Labor-unabhängige Workflow-Definitionen, deren gerätebezogene Aktivitäten für alle Geräte einer Geräteklasse allgemein gültig sind.

Mit Hilfe des beschriebenen DCMS wird die Interaktion zwischen den Komponenten des GHRC-Prototyps mit Laborgeräten[259], Barcode-Equipment und Dockingstationen realisiert.

5.5.2.2 Der *DCMSFlashRemoteServer*

Exemplarisch für die Anbindung von Geräten mit Hilfe des DCMS an die konzipierten Systemkomponenten wird im Folgenden die Anbindung der Dockingstationen (siehe Abschnitt 5.4.3) gezeigt[260]. Zum besseren Verständnis wird zunächst der elektronische Aufbau von Trägerplatten und Lagerboxen beschrieben und darauf basierend Funktion und Komponenten der Dockingstationen motiviert. Anschließend werden exemplarisch die DCUnit *DCMSFlashServer* zur Steuerung der realisierten Docking-Stationen, ihr Kommandosatz und ihr Parametersatz beschrieben.

Wie in Kapitel 4.3 geschildert, enthalten die Sockel von ‚Icebreaker' und ‚Portlink' jeweils einen seriellen Flash-Speicherchip. Die serielle Schnittstelle entspricht dem *Serial Peripheral Interface*-Standard[261] (*SPI*). Somit kann ein Datenaustausch zwischen den seriellen Flash-Speicherchips und einem Computer mittels eines *SPI-Hostadapters* stattfinden. In Kapitel 4.3 wurden Lagerboxen beschrieben, die der Organisation von Proben dienen und als Cryo-Devices Teil der elektronischen Lagertank-Infrastruktur (siehe Kapitel 4.4) sind. Zur

---

[259] Die Integration von Laborgeräten und Geräten der in Kapitel 4.4 beschriebenen Technologieplattform findet gegenwärtig statt und ist noch nicht vollständig abgeschlossen. Bereits zum großen Teil fertiggestellt sind die Integration des Kryolagerturms, der Kryowerkbank und des elektronischen Inventars. Diese Geräte können bereits aus Workflow-Definitionen heraus unter Verwendung des DCMS und gerätespezifischer DCUnits und entsprechender DCJobs angesteuert werden.

[260] Grund: die Docking-Station ist ein von allen Systemkomponenten verwendetes Gerät.

[261] SPI realisiert eine einfache Kommunikation auf Basis von Datenströmen, die keinem vordefinierten Protokoll genügen muß.

Serienreife entwickelt wurden Lagerboxen mit 24 Steckverbindern, prototypisch existieren auch Lagerboxen zur Aufnahme von 96 diskreten Substraten mit integriertem seriellem Flash-Speicher. Alle Lagerboxen enthalten ein eigenes PCB-Board mit jeweils vier fest verlöteten SPI-Flash-Chips, die den gemeinsamen Speicher der Lagerbox darstellen. Dieser steht unabhängig von aufgesteckten Substraten zur Verfügung. Insgesamt muß also die selektive Ansteuerung jeweils eines von maximal 100 Flash-Speicherchips möglich sein. SPI erfordert zusätzlich zur Spannungsversorgung die folgenden vier Signale: Clock (SCLK), Master Output/Slave Input (MOSI), Master Input/Slave Output (MISO) und ChipSelect. Damit ein SPI-Flash-Chip gelesen oder beschrieben werden kann, müssen simultan ChipSelect und Clock anliegen.

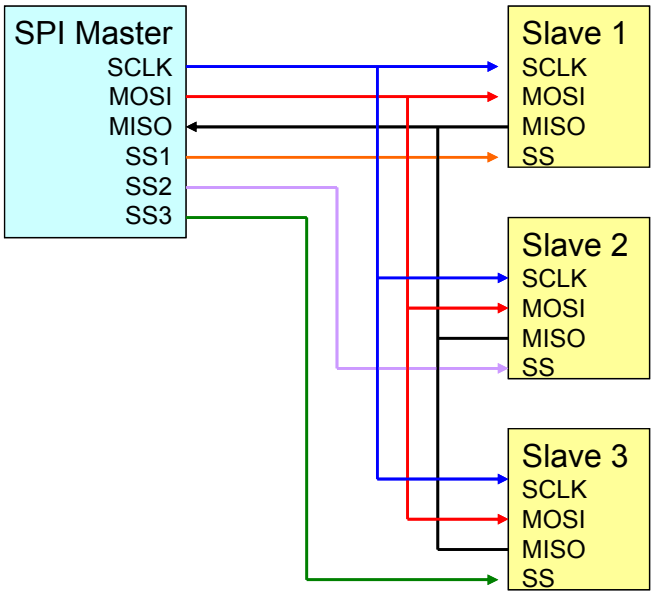

Abbildung 60: SPI-Master-Slave-Anordnung am Beispiel eines SPI-Host-Adapters (Total Phase ‚Cheetah') und drei serieller Flash-Speicherchips. Der Host-Adapter ist blau dargestellt und verfügt über 3 ChipSelect-Leitungen (SS1, SS2, SS3). Die seriellen Flash-Speicherchips sind gelb dargestellt. Bei direktem Anschluß wie in dieser Anordnung kann der Host-Adapter maximal aus drei Flash-Speicherchips selektieren. Die Datenleitungen können parallel miteinander verbunden sein. Ein lesender oder schreibender Zugriff auf einen SPI-Flash-Speicherchip erfordert das gleichzeitige Anliegen des ChipSelect-Signals und des Clock-Signals an den Pins des Speicherchips.

Eine effiziente Methode zur Selektion aus mehr als drei Speicherchips wird in [Shirley 2006] konzipiert und erfordert die Organisation der Steckplätze der Lagerboxen in Zeilen und Spalten einer 10x10-Matrix. Eine geeignete Selektionslogik besteht im Wesentlichen aus zwei Demultiplexern; einer davon routet das Clock-Signal zur selektierten Zeile der Matrix, der andere routet das ChipSelect-Signal zur selektierten Spalte. Nur derjenige Speicherchip, an dem gleichzeitig beide Signale anliegen, ist funktional. Nach Adressierung der Spalten- und Zeilenadresse verhält sich die Matrix aus seriellen Flash-Speicherchips bzw. Steckplätzen für diskrete Substrate wie ein einziger Speicherchip, so daß der selektierte Chip gelesen oder beschrieben werden kann. Die folgende Abbildung (entnommen aus [Shirley 2006] veranschaulicht diese Selektionsmethodik:

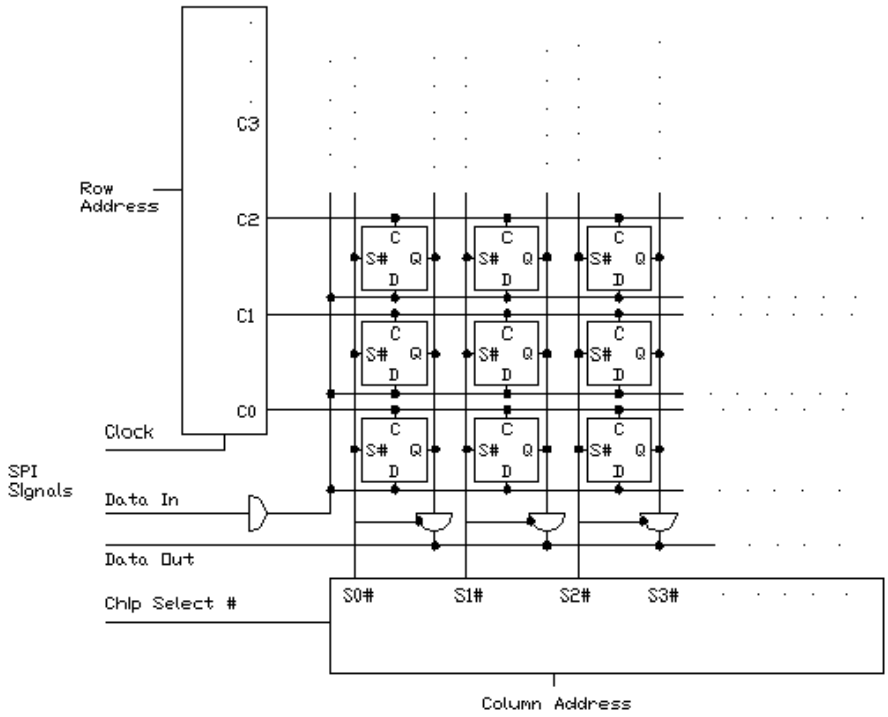

Abbildung 61: Schema einer effizienten Steckplatz-Selektion (entnommen aus [Shirley 2006]), basierend auf der Verwendung von zwei Demultiplexern. Die Steckverbinder für die seriellen Flash-Speicherchips diskreter Substrate werden in einer 10x10-Matrix angeordnet. Jeweils ein Demultiplexer routet das Clock-Signal zur ausgewählten Zeile bzw. zur ausgewählten Spalte der Matrix. Derjenige Steckplatz, an dem beide Signale simultan anliegen, erlaubt lesenden oder schreibenden Zugriff auf den eingesteckten Speicherchip. Es sind zusätzliche Signale zur Steuerung der Ausgabe der Demultiplexer auf die gewünschte Zeile bzw. Spalte der Matrix erforderlich.

Um eine wie oben dargestellte Selektion durch eine externe Schaltung auszuführen[262], die nicht Teil des PCB-Boards der Lagerbox ist, sondern nach Bedarf mit der Platine verbunden werden kann, müssen die Leitungen der Matrix über einen Steckverbinder nach außen geführt werden. Im Fall der oben beschriebenen Lagerboxen erfolgt dies mittels eines 25-poligen D-Sub-Steckverbinders, der somit unter anderem die zehn ChipSelect-Leitungen und die zehn Clock-Leitungen der Matrix nach außen führt[263]. Eine externe Schaltung, die mittels der herausgeführten ChipSelect- und Clock-Leitungen des PCB-Boards der Lagerbox die Selektion eines bestimmten Steckplatzes anhand der oben beschriebenen Methodik ermöglicht, muß somit in den Dockingstationen (siehe Abschnitt 5.4.3) integriert werden. Zusätzlich muß für die Kommunikation zwischen selektiertem Chip und Applikation ein SPI-Bus etabliert werden. Deshalb wurde in die Dockingstationen auch ein SPI-Host-Adapter[264] integriert, der einerseits einen SPI-Bus etabliert, die Selektionsschaltung mittels SPI-Signalen steuert und

[262] Diese Vorgehensweise wurde motiviert durch die Notwendigkeit, möglichst wenige aktive elekronische Bauelemente in die Lagerboxen zu integrieren.

[263] außerdem Vcc, GND, /WP, D und MISO

[264] Total Phase ‘Cheetah’ USB-SPI-Hostadapter

Daten mit dem selektierten Chip austauscht. Unter anderem wird ein 8-bit serial-in Shift-Register mit acht parallelen Ausgängen eingesetzt, um jeweils ein Byte vom MOSI-Ausgang des Host-Adapters in Eingangssignale für 3-to-8-line Decoder und für 4-to-16-line Decoder zu verwenden. Über diese Bausteine werden die ChipSelect-Signale und das Clock-Signal auf die herausgeführten ChipSelect-Leitungen und Clock-Leitungen des PCB-Boards der Lagerbox aufgeschaltet und somit einer der Chips der Lagerbox selektiert. Folgende Tabelle zeigt exemplarisch die Zuordnung zwischen Chipnummer, Byte und Position auf einem PCB-Board mit vier gemeinsamen, fest verlöteten SPI-Speicherchips und 24 Steckverbindern für diskrete Substrate[265]:

Chipnummer	Byte	Position	Chipnummer	Byte	Position
1	0x00	0	15	0x24	A6
2	0x01	1	26	0x05	B1
3	0x02	2	27	0x06	C1
4	0x03	3	28	0x07	D1
5	0x04	A1	29	0x08	A2
6	0x10	C2	30	0x09	B2
7	0x11	D2	31	0x15	D3
8	0x12	A3	32	0x16	A4
9	0x13	B3	33	0x17	B4
10	0x14	C3	34	0x18	C4
11	0x20	A5	35	0x19	D4
12	0x21	B5	36	0x25	B6
13	0x22	C5	37	0x26	C6
14	0x23	D5	38	0x27	D6

Tabelle 11: Übersicht über die erforderlichen Bytes zur Selektion eines seriellen Flash-Speicherchips auf einem PCB-Board einer Lagerbox mit vier fest verlöteten Speicherchips und 24 Steckplätzen für diskrete Substrate. Das erforderliche Byte für die Selektion wird anhand der gewünschten Position ausgewählt und über den in der Dockingstation integrierten SPI-Host-Adapter auf die Eingänge der Selektionslogik der Dockingstation ausgegeben. Die entsprechenden Demultiplexer werden durch das Byte gesteuert und schalten das ChipSelect-Signal bzw. das Clock-Signal auf die entsprechende Zeile der 10x10-Matrix des PCB-Boards der Lagerbox auf. Damit wird die gewünschte Position funktional.

Die Interaktion der Systemkomponenten (Stationensoftware, Formatierungsapplikation, Initialisierungsapplikation) mit den oben beschriebenen Docking-Stationen wurde mit Hilfe des beschriebenen DCMS realisiert. Zu diesem Zweck wurde für die Steuerung der Dockingstationen die DCUnit *DCMSFlashRemoteServer* implementiert. DCMSFlashRemoteServer kommuniziert wie in Abbildung 59 illustriert über TCP/IP-Socketverbindungen mit dem

[265] Die in Abschnitt 5.4.3. konzipierten Trägerplatten für Standard-Laborröhrchen, die zum Austausch der Informationen zwischen Laborstationen benötigt werden, enthalten die gleichen PCB-Boards. Diese sind jedoch nur mit den gemeinsamen Speicherchips ausgestattet und verfügen nicht über Steckverbinder für diskrete Substrate. Die Dockingstationen sind daher mit Lagerboxen und Trägerplatten kompatibel.

Commander, also der jeweiligen Systemkomponente des GHRC-Prototyps, und steuert die Selektion eines gewünschten seriellen Flash-Speicherchips auf einer Lagerbox oder Trägerplatte anhand der oben beschriebenen Methodik unter Verwendung der Selektionsbytes in Tabelle 11[266]. Nach erfolgter Selektion ist der Datenaustausch mit dem jeweiligen Chip möglich[267]. Die erforderlichen Kommandos zur Steuerung der Dockingstation wurden in Form von DCJobs implementiert und stellen den Kommandosatz des DCMSFlashRemoteServers dar. Sie stehen innerhalb des DCMS zur Verfügung und können vom jeweiligen Commander[268] aufgerufen werden:

Kommando / DCJob	**Parameter**	**Funktion**
SelectFlash ()	byte	Selektiert einen Steckplatz anhand der oben beschriebenen Selektionsmethodik unter Verwendung eines Steckplatz-spezifischen Bytes gemäß obiger Tabelle[269] und setzt chipSelect
IsChipAvailable ()	chipSelect	Prüft, ob im selektierten Steckplatz ein Chip eingesteckt ist
GetChipInfo ()	chipSelect	liest den *JEDEC*[270]-ID Code aus dem SPI-Flash-Speicher aus (Manufacturer, MemID1, MemID2)
GetStatusRegister ()	chipSelect	Liest das Status-Register aus
SetStatusRegister ()	chipSelect	Setzt das Status-Register
Power ()	Convert.ToByte(1)	Einschalten der Spannungsversorgung der Demultiplexerplatine
ReadFlash ()	chipSelect area	Lesen des Inhalts des ausgewählten Chipbereichs (‚Protected', ‚Zipfile only') auf dem gewählten Speicherchip[271]
ReadPage ()	page chipSelect	Lesen des Inhalts einer ausgewählten Speicherseite
EraseFlash ()	chipSelect	Löschen des ausgewählten Chipbereichs (‚Protected', ‚Zipfile only')
WritePage ()	Page value chipSelect	Beschreiben einer bestimmten Speicherseite im selektierten Flash
WriteFlash ()	User File Pages ChipSelect area	Beschreiben des ausgewählten Chipbereichs (‚Protected', ‚Zipfile only') mit binären Daten

[266] Die Tabelle ist als Dictionary in den DCMSFlashRemoteServer integriert.

[267] Der DCMSFlashServer kommuniziert via USB mit dem SPI-Host-Adapter. Der Zugriff auf die Cheetah-Funktionalität wird durch Funktionsaufrufe an die Cheetah API ermöglicht.

[268] Als Commander können Workflow-Engines oder GUI-Applikationen agieren.

[269] Dieser DCJob kapselt folgende einzelne Befehle der Cheetah API: ‚EnableOutput', ‚SelectChip', ‚ShiftBytes', ‚EnableOutput'.

[270] *JEDEC*: 'Joint Electron Device Engineering Council', mittlerweile 'Solid State Technology Association'.

[271] Die Position der Bereiche auf dem Speicherchip wird im Index der Datenstruktur definiert (siehe Kapitel 5.6)

CreateData ()	DevType chipSelect	Kreiert anhand des ausgewählten DeviceTypes (siehe Kapitel 5.6) eine Instanz des für den ausgewählten Chip des ‚Admin'-Bereichs definierten Teils der Datenstruktur für dezentrale Wissensbewahrung und Cryo-Devices.

Tabelle 12: DCJobs der DCUnit *DCMSFlashRemoteServer*. Die aufgelisteten DCJobs stellen den Kommandosatz des DCMSFlashRemoteServers dar, der innerhalb des DCMS von einem Commander aufgerufen werden kann. Bei den abgebildeten DCJobs handelt es sich um modulare, gekapselte Kombinationen von Befehlen der Cheetah API, die zur Interaktion der Systemkomponenten mit den Dockingstationen und zum Beschreiben oder Lesen einzelner gemeinsamer Chips von Trägerplatten oder Lagerboxen oder aufgesteckter diskreter Substrate benötigt werden. Enthalten sind auch die DCJobs zur Ansteuerung der in der Dockingstation enthaltenen Demultiplexer, die zur Selektion des jeweiligen seriellen Flash-Speicherchips anhand der oben beschriebenen Selektionsmethodik benötigt werden.

Das Service-GUI des DCMSFlashRemoteServers ist nachfolgend illustriert. Es erlaubt den Aufruf der beschriebenen DCJobs auch unabhängig von der Steuerung durch einen Commander:

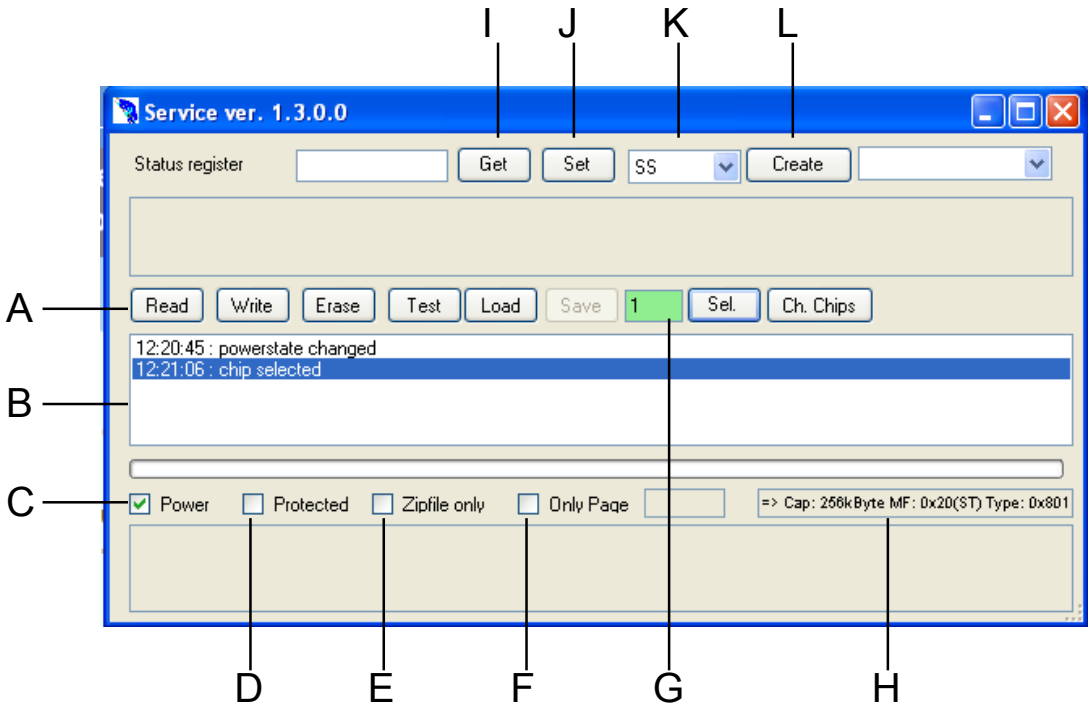

Abbildung 62: Das Service-GUI des DCMS-FlashRemoteServers. Unter seiner Verwendung lassen sich die Dockingstationen auch unabhängig von einem Commander, beispielsweise einer GUI-Applikation oder einer Workflow-Engine, steuern und lesende und schreibende Zugriffe auf die gemeinsamen Speicherchips von Lagerboxen oder Trägerplatten oder auf die Speicherchips von ‚Icebreakern' oder ‚Portlinks' ausführen. (A) Schaltflächen zum direkten Aufrufen der DCJobs ‚ReadFlash()', ‚ReadPage()', ‚WriteFlash()', ‚WritePage()', ‚EraseFlash()'(seitenweise Funktion wird durch Schaltfläche (F) gewählt). Weitere Schaltflächen zum Testen des gelesenen Inhalts auf Übereinstimmung mit vordefinierten Strukturen, zum Laden von Dateien von der lokalen Festplatte und zum Aufruf des DCJobs ‚SelectFlash()'. (B) Informationsbereich für Statusmeldungen. (C) Schaltfläche zum Einschalten der Spannungsversorgung der Selektionsschaltung in den Dockingstationen durch Aufruf des DCJobs ‚Power()'. (D) Checkbox zur Auswahl des Protected Area des Speicherchips beim Verwenden der in (A) beschriebenen Schaltflächen. (E) Checkbox zur Auswahl des User-Areas (.zip-Struktur) des Speicherchips

beim Verwenden der in (A) beschriebenen Schaltflächen. (F) Checkbox zum Umschalten der in (A) beschriebenen DCJobs auf seitenweise Funktion. (G) Eingabefeld für die gewünschte Chipnummer. (H) Anzeigefeld für die anhand der JEDEC-ID festgestellten Chipmerkmale (Hersteller, Kapazität, Typ). (I), (J) Schaltflächen zum Lesen oder Setzen des Status-Registers durch Aufruf der DCJobs ‚GetStatusRegister()' oder ‚SetStatusRegister()'. (K) Kombinationsfeld zur Auswahl einer der vordefinierten Datenstrukturen zum Formatieren des jeweiligen Cryo-Devices (siehe Kapitel 5.6). (L) Schaltfläche zum Instanziieren der Datenstruktur und Formatieren des Cryo-Devices durch den DCJOB ‚CreateData()'.

Jede Instanz der Stationensoftware, der Initialisierungsapplikation und der Formatierungsapplikation ist mit einer Dockingstation und einer Instanz des DCMSFlashRemoteServers ausgestattet. Weitere häufig benötigte Geräte[272] an Laborstationen sind Barcodescanner und Barcodedrucker für die effiziente Identifizierung von Proben und das Ausdrucken von Etiketten (‚Label-on-Demand', siehe Abschnitt 5.4.4).

5.5.3 Die Systemkomponenten

Für das Handling der Containerdatei, die durch den Workflow-Editor erzeugt wird (siehe Abschnitt 5.5.1.4) und für die Initialisierung, Bearbeitung und Abarbeitung ihrer Komponenten (beispielsweise XML-Workflow-Markup und Target-File) wurden gemäß der Konzeption des GHRC-Prototyps folgende Applikationen implementiert:

- Formatierungsapplikation: ‚ChameleonManufacturingStation'
- Initialisierungsapplikation: ‚ChameleonInitialStation'
- Stationensoftware: ‚ChameleonStation'
- Ansichts- und Exportapplikation: ‚ContentViewer'

5.5.3.1 Die Formatierungsapplikation

Auch für den Einsatz bei der Serienproduktion der Cryo-Devices konzipiert, ermöglicht die Formatierungsapplikation ‚ChameleonManufacturingStation' die Formatierung von Trägerpatten, Lagerboxen und einzelnen Substraten mit integriertem Speicherchip. Sie stellt dazu ein einfaches GUI zur Verfügung, um die für eine Formatierung erforderlichen DCJobs des DCMSFlashRemoteServers aufzurufen. Diese DCJobs steuern das Verhalten einer angeschlossenen Dockingstation. Im Sinne der Abbildung der Hierarchieebenen des DCMS in Abschnitt 5.5.2 stellt die ‚ChameleonManufacturingStation' den Commander dar.

---

[272] Für jedes Gerät, das von den Systemkomponenten gesteuert werden soll, ist eine gerätespezifische Instanz der Klasse DCUnit erforderlich.

Nach Identifikation der gemeinsamen seriellen Flash-Speicherchips von Trägerplatte oder Lagerbox durch den DCJob ‚GetChipInfo()' kann eines von mehreren vordefinierten[273] Trägerplatten- bzw. Lagerboxenlayouts gewählt werden. Die Layouts unterscheiden sich hinsichtlich Anzahl und Anordnung der Steckplätze für Röhrchen und hinsichtlich der Unterstützung diskreter Substrate mit integriertem Speicherchip. Grundsätzlich werden bei der Formatierung unterschiedliche Layouts der beiden folgenden Cryo-Device-Typen[274] (siehe auch Kapitel 5.6) unterstützt:

- Trägerplatten für Standardröhrchen (ohne Chip, Datenstruktur ‚CDv_MultiSample')
- Lagerboxen für ‚Icebreaker' oder ‚Portlinks' (Datenstruktur ‚CDv_SampleParent')

Nach Auswahl von Layout und Cryo-Device-Typ formatiert ‚ChameleonManufacturingStation' das Cryo-Device. Eine Instanz der entsprechenden Datenstruktur wird in die gemeinsamen Flash-Speicherchips (‚Admin-Bereich[275]') des Cryo-Device geschrieben. Zu diesem Zweck wird für jeden dieser Speicherchips durch Aufruf des DCJobs ‚CreateData()' des DCMSFlashRemoteServers der benötigte Teil der Datenstruktur instanziiert. Zusätzlich zur Instanziierung der Datenstruktur werden die Informationen über gewählte Layout in Form einer XML-Datei (‚PlateDesc.xml') im User-Area der kreierten Datenstruktur abgelegt. Initialisierungsapplikation sowie Stationensoftware adaptieren ihr GUI und ihre Funktion anhand des Inhalts dieser Datei (siehe auch Abschnitt 5.4.4 und 5.4.5). Wenn lediglich einzelne ‚Icebreaker' oder ‚Portlinks' mittels einer bereits formatierten Lagerbox formatiert werden sollen, werden die einzelnen Steckplätze der Lagerbox nacheinander selektiert und für jedes diskrete Substrat eine Instanz der Datenstruktur ‚CDv_SingleSample' instanziiert und auf den jeweils selektierten Speicherchip geschrieben.

5.5.3.2 Die Initialisierungsapplikation

Die Initialisierungsapplikation ‚ChameleonInitialStation' wird zur Auswahl der benötigten Workflow-Definition[276] verwendet, zum Erstellen der initialen ToDo-Liste[277] innerhalb des Target-Files anhand der zu bearbeitenden Proben und zum Speichern der modifizierten Containerdatei innerhalb des User-Areas der Datenstruktur von Trägerplatte, Lagerbox oder diskreten Substraten. Zu diesem Zweck steuert sie die Interaktion der Dockingstationen mit den genannten Cryo-Devices. Nach der Initialisierung enthält die Trägerplatte unter anderem

---

[273] ManufacturingStation erlaubt die Erweiterung dieser Auswahl um neue Cryo-Device-Typen und Layouts.

[274] Die Kompatibilität der Cryo-Devices mit der elektronischen Lagertank-Infrastruktur (Kapitel 4.4) wird durch die Formatierung mit der entsprechenden Datenstruktur hergestellt.

[275] Der ‚Admin-Bereich' entspricht den vier fest verlöteten seriellen Flash-Speicherchips des PCB-Boards.

[276] Diese liegt in Form einer Containerdatei (siehe Abschnitt 5.5.1.4) in einem Repository vor.

[277] Siehe auch Abschnitt 5.4.5.

den selektierten Workflow und ein instanziiertes Target-File, das Aufschluß über die anhand der Workflow-Definition zu präparierenden Proben und die Anzahl ihrer Aliquoten gibt. Die ‚ChameleonInitialStation' schafft somit die Ausgangsposition für die in Abschnitt 5.4.3 beschriebene Interaktion der Laborstationen mit Trägerplatten und Lagerboxen.

Zur effizienten Registrierung zu bearbeitender Proben ist die ‚ChameleonInitialStation' mit einem Barcodescanner ausgestattet. Zusätzlich verfügt sie über einen Barcodedrucker, der es erlaubt, handschriftlich gekennzeichnete Proben mit einem standardisierten Etikett zu versehen, das bei der anschließenden Bearbeitung der Probe eine effiziente Identifizierung innerhalb des in Abschnitt 5.4.4 konzipierten Proben- und Aliquoten-Managements ermöglicht. Barcodescanner und Barcodedrucker sind über gerätespezifische Instanzen der Klasse DCUnit (siehe Abschnitt 5.2.2) in das DCMS integriert und werden durch entsprechende DCJobs gesteuert. Die folgende Abbildung zeigt schematisch den Aufbau und die Ausstattung der ‚ChameleonInitialStation':

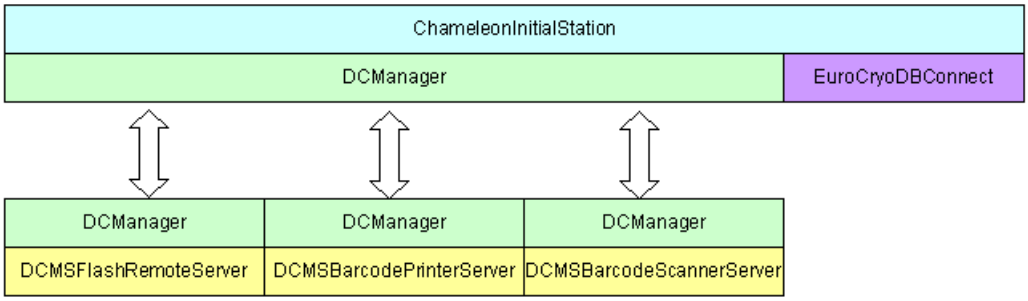

Abbildung 63: Schematischer Aufbau der ‚ChameleonInitialStation' als GUI-Applikation, die über das DCMS mit angeschlossenen Geräten interagiert. Sie stellt innerhalb der DCMS-Hierarchie den Commander dar und kommuniziert mit Hilfe zweigeteilter DCManager via TCP/IP Sockets mit den DCUnits von Barcodescanner, Barcodedrucker und mit dem oben beschriebenen DCMSFlashRemoteServer zur Interaktion mit Trägerplatten, Lagerboxen und diskreten Substraten. Aus Gründen der Robustheit laufen alle DCUnits als eigene Prozesse ab. Während der Registrierung von Proben kann deklaratives Probenwissen per Webservice aus einer Probendatenbank (hier: ‚EurocryoDB') angefordert und im User-Area der Datenstruktur auf den Cryo-Devices abgelegt werden.

GUI und Verhalten der ‚ChameleonInitialStation' passen sich dem Layout des mit der Dockingstation verbundenen Cryo-Devices an. Dies erfolgt anhand der im User-Area gespeicherten XML-Datei ‚PlateDesc.xml' (siehe Abschnitt 5.5.3.1). Der jeweils nächste zu belegende Steckplatz des Cryo-Device wird im GUI der ‚ChameleonInitialStation' visuell hervorgehoben. Schreibvorgänge auf den Speicherchip eines Röhrchens werden ebenfalls visuell angezeigt. Folgende Abbildung zeigt das GUI der ‚ChameleonInitialStation' während der Registrierung von Proben:

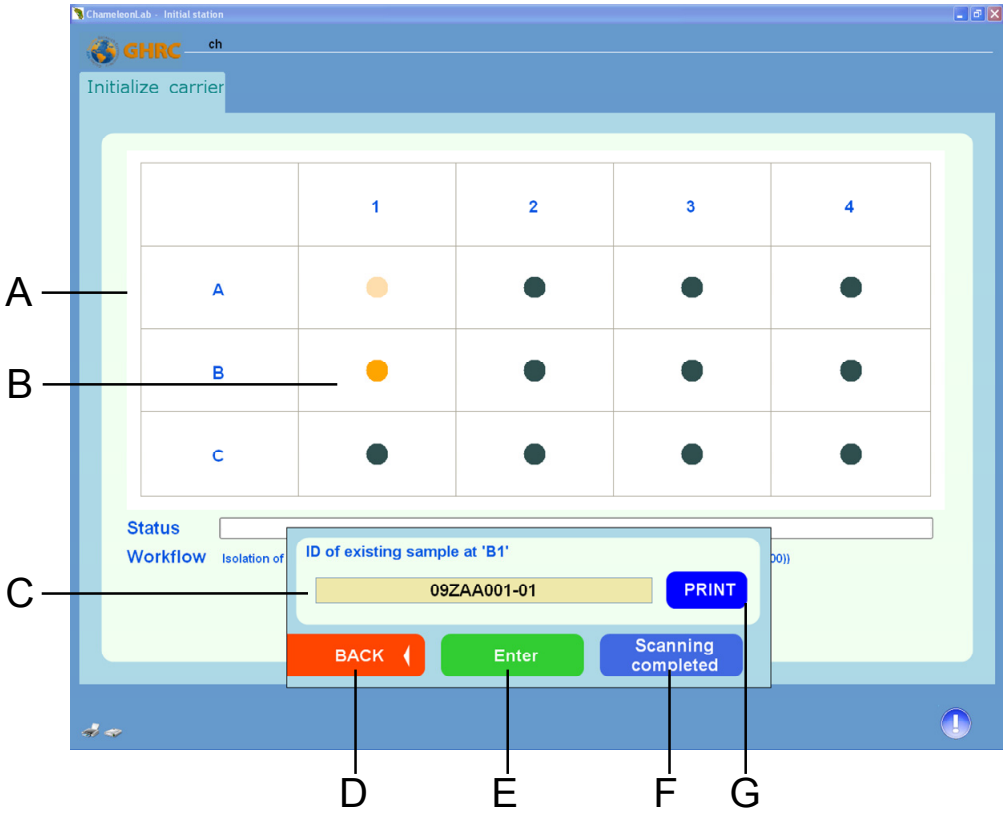

Abbildung 64: Das GUI der ‚ChameleonInitialStation' während der Registrierung von Proben. Die Auswahl eines Workflows ist zu diesem Zeitpunkt bereits erfolgt. Das PopUp zur Probenregistrierung verdeckt im Bild teilweise die Beschreibung des ausgewählten Workflows. (A) Symbolische Darstellung des Layouts des aufgesteckten Cryo-Devices, hier einer Trägerplatte für Standardröhrchen. Die Steckplätze für einzelne Probenröhrchen sind in Zeilen und Spalten angeordnet, hier in Form einer 4x3-Matrix. Das GUI liest zum Zweck der Adaption eine Datei aus dem User-Area des Cryo-Devices aus, das Layout-Informationen enthält. (B) die nächste zu füllende Position des Cryo-Devices wird durch farbliche Akzentuierung angezeigt. Blassgelbe Farbe zeigt belegte Positionen an, orange Farbe zeigt den aktuellen Steckplatz an, dunkelgraue Farbe zeigt leere Steckplätze an. (C) PopUp zur Probenregistrierung. Die ProbenID kann sowohl per Tastatur in das Eingabefeld eingegeben werden als auch per Barcodescanner. (D) Schaltfläche zum Verlassen des PopUps ohne Probenregistrierung. (E) Schaltfläche zur Bestätigung der aktuellen ProbenID und zum Beginnen der Registrierung der nächsten Probe. (F) Schaltfläche zum endgültigen Abschließen der Probenregistrierung. (G) Schaltfläche zum Ausdrucken von Barcodeetiketten mit Hilfe der DCUnit DCMSBarcodePrinterServer, beispielsweise für handbeschriftete Probenröhrchen.

5.5.3.3 Die Stationensoftware

Die in Abschnitt 5.4.4 erarbeiteten Konzepte für eine Stationensoftware wurden in Form der ‚ChameleonStation' implementiert. Im Unterschied zu ‚ChameleonInitialStation' und ‚ChameleonManufacturingStation' wird die Interaktion mit Geräten nicht nur durch GUI-Events ausgelöst, sondern auch durch die Abarbeitung entsprechender Meta-Aktivitäten in der Workflow-Definition durch die Workflow-Engine. Die in Meta-Aktivitäten enthaltenen Meta-Kommandos und Meta-Parameter werden wie in Abschnitt 5.5.2.1 beschrieben durch Interaktion der Hierarchiebenenen des DCMS ausgeführt. Die Abarbeitung einer Workflow-Definition durch ‚ChameleonStation' setzt das Einlesen der Containerdatei (siehe Abschnitt 5.5.1.4)

von initialisierten Trägerplatten oder Lagerboxen voraus. Dazu erforderlich ist die Integration einer Dockingstation in die jeweilige Instanz der ‚ChameleonStation', die mit Hilfe des DCMS (siehe Abschnitt 5.5.2) und einer Instanz des DCMSFlashRemoteServers (siehe Abschnitt 5.5.2.2) realisiert wird.

Das Einlesen der Containerdatei von einem initialisierten Cryo-Device wird durch ein GUI-Event ausgelöst (Betätigen der ‚Open'-Schaltfläche). Anhand des Index der Datenstruktur (siehe Kapitel 5.6) kann die Position des User-Areas auf den Flash-Speicherchips der Trägerplatte oder Lagerbox bestimmt werden, so daß mit Hilfe des DCJobs ‚ReadFlash()' (siehe Abschnitt 5.5.2.2) gezielt die Containerdatei aus der Datenstruktur extrahiert und in den Cache der ‚ChameleonStation' übertragen werden kann. Die Containerdatei war zuvor von der ‚ChameleonInitialStation' an diese Position geschrieben worden (siehe Abschnitt 5.5.3.2), nachdem das Workflow-spezifische Target-File-Template um die initiale ToDo-Liste ergänzt worden war. Auf ähnliche Weise kann eine Containerdatei auch aus dem Flash-Speicherchip eines einzelnen Substrates eingelesen werden.

Während der Abarbeitung der Workflow-Definition durch eine oder mehrere Instanzen von ‚ChameleonStation' (verschiedenen Laborstationen entsprechend) wird das Target-File gemäß seiner Struktur um anfallende Dokumentationsdaten, Zeitstempel und Informationen des Proben- und Aliquoten-Managements (siehe Abschnitt 5.4.4) ergänzt.

Das Zurückschreiben der Containerdatei aus dem Cache einer Instanz von ‚ChameleonStation' in das User-Area des Admin-Bereichs eines Cryo-Devices erfolgt durch den DCJob ‚WriteFlash()' des DCMSFlashRemoteServers. Dieser Vorgang wird durch Betätigen der GUI-Schaltfläche ‚RemoveCarrier' ausgelöst. Dieser Vorgang ist dann erforderlich, wenn die Abarbeitung der Workflow-Definition an einer anderen Laborstation fortgeführt werden soll (siehe auch Abschnitt 5.4.3). Zusätzlich zum Aufruf von ‚WriteFlash()' wird eine Kopie des Stationencaches in ein allen Stationen zugängliches gemeinsames Verzeichnis kopiert und das aktuelle Target-File per Webervice an die Probendatenbank übergeben. Die Datenbank kann durch Auswertung des Target-Files ein Präparationstracking durchführen. Ein ähnlicher Vorgang wird durch die Schaltfläche ‚ChangeCarrier' ausgelöst, die dem Umkopieren der Informationen einer Trägerplatte auf eine Trägerplatte mit anderem Layout dient[278].

Als Commander im Sinne des DCMS steuert ‚ChameleonStation' eine modular erweiterbare (siehe Abschnitt 5.5.2) Ausstattung an Geräten mittels spezifischer DCUnits. Zusätzlich zur oben motivierten Dockingstation sind als weitere Bestandteile der Standard-Ausstattung einer ‚ChameleonStation' ein Barcodescanner (DCUnit: *DCMSBarcodeScannerServer*) und ein Barcodedrucker (DCUnit: *DCMSBarcodePrinterServer*) zu nennen, gemäß der fest-

[278] Ein Trägerplattenwechsel wird dann notwendig, wenn während der Präparation Umfüllvorgänge in Probengefäße mit anderem Formfaktor stattfinden.

gestellten ‚impliziten' Anforderungen an effizientes Identifizieren und flexibles Etikettieren von Probenröhrchen (siehe auch Abschnitt 5.3.3.4). Die DCJobs dieser DCUnits können durch GUI-Events (beispielsweise durch die Schaltfläche ‚PrintLabel') aufgerufen werden. In diesem Zusammenhang erwähnenswert ist die Möglichkeit, schwierig verbalisierbares Wissen (beispielsweise die Verteilung der Phasen nach einer Dichtezentrifugation) mittels einer Fotofunktion zu akquirieren, die in Form eines DCJobs des DCMSBarcodeScannerServers implementiert wurde. Der minimale Aufbau einer ‚ChameleonStation' ist in folgender Abbildung schematisch dargestellt:

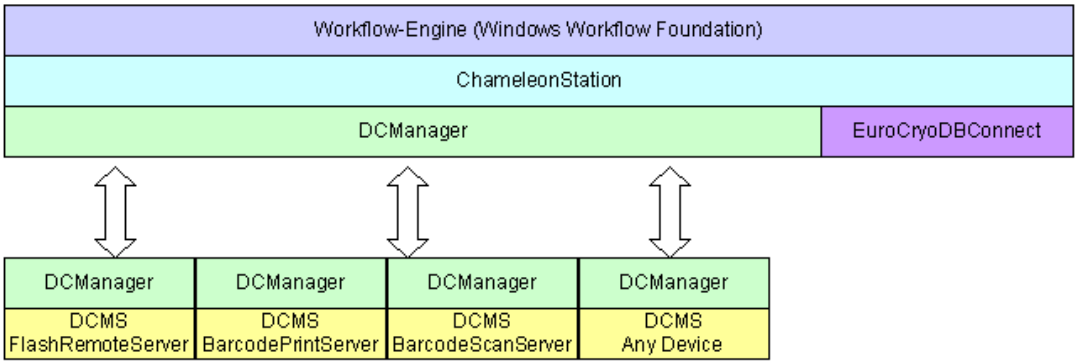

Abbildung 65: Minimaler schematischer Aufbau einer ChameleonStation. Zusätzlich zu den bei der ‚ChameleonInitialStation' beschriebenen Elementen ist hier die Integration einer Workflow-Engine bedeutsam, die mit der ChameleonStation interagiert und mit Hilfe von Meta-Aktivitäten in der Workflow-Definitionen auch DCJobs angeschlossener DCUnits auslösen kann. Als Commander innerhalb des DCMS agieren hier sowohl die GUI-Applikation wie auch die integrierte Workflow-Engine. Die Geräteausstattung einer ‚ChameleonStation' ist modular um weitere gerätespezifische DCUnits erweiterbar. Ihre spezifischen DCJobs stehen dann für weitere Meta-Aktivitäten in der Workflow-Definition zur Verfügung. Die Kompatibilität einer Laborstation mit gerätebezogenen Meta-Aktivitäten kann somit durch zusätzliche DCUnits und deren Kommandosätzen hergestellt werden. Die DCUnits laufen in eingenen Prozessen ab und kommunizieren über TCP/IP Sockets mit dem DCManager der ‚ChameleonStation'.

Das GUI der ‚ChameleonStation' wurde gemäß der Konzeption in Abschnitt 5.4.6.3 gestaltet und implementiert. Ausgabebereich, Status-Bereich und GUI-Plugins (für komplexe Code-Aktivitäten oder Geräteklassen) werden für jede Workflow-Aktivität auf jeweils einer eigenen Registerkarte dargestellt. Alle Workflow-Aktivitäten werden direkt nach dem Laden der Containerdatei auf den Registerkarten angeordnet, so daß ein Blättern durch die gesamte Präparation möglich ist. Folgende Abbildung illustriert und beschreibt das GUI der ‚ChameleonStation':

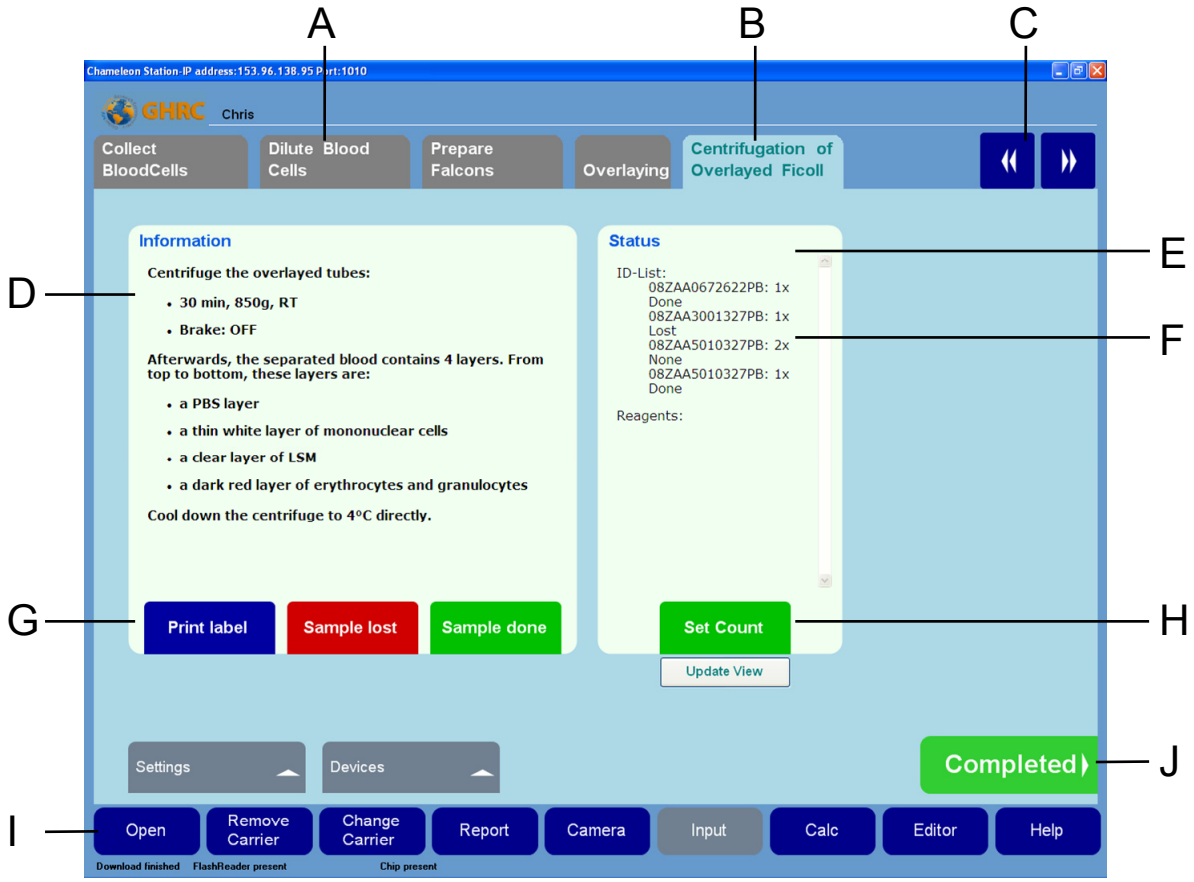

Abbildung 66: Das GUI der ‚ChameleonStation', erstellt auf Basis der Konzeption in Abschnitt 5.4.6.3. Unmittelbar nach dem Laden der Containerdatei mit der Workflow-Definition werden alle Anwender-bezogenen Aktivitäten innerhalb einer Registeranordnung platziert. (A) Jede Aktivität wird auf jeweils einer eigenen Registerkarte dargestellt. Die Register werden dynamisch anhand der Workflow-Definition beschriftet, so dass eine Orientierung anhand der Registernamen möglich wird. Inaktive oder bereits abgeschlossene Aktivitäten werden durch eine graue Registerlasche symbolisiert. (B) Die aktuell auszuführende Aktivität wird durch eine hellblaue Registerkarte symbolisiert. Ein Blättern durch die gesamte Workflow-Definition wird durch Navigationsschaltflächen (C) ermöglicht, um dem Anwender eine Orientierung zu ermöglichen und einen Überblick über benötigte Ressourcen zu geben. (D) Auf jeder Registerkarte vorhanden ist ein Ausgabebereich, der das prozedurale Wissen zur Ausführung der aktuellen Aktivität explizit und detailliert zur Verfügung stellt, um die Substitution lückenhafter Protokolle durch erfahrungsgeleitetes Arbeitshandeln auf Basis individuellen impliziten Wissens zu minimieren. Alle Registerkarten zusammen stellen das explizite prozedurale Wissen eines bestimmten Protokolls dar. Im Ausgabebereich können statt prozeduralem Wissen auch komplexe gerätespezifische GUI-Plugins dargestellt werden, die über den Betriebszustand von Laborgeräten informieren, die aus Meta-Aktivitäten heraus angesteuert werden. Auch komplexe Code-Aktivitäten können mit Hilfe von GUI-Plugins dargestellt werden. (E) Auf jeder Registerkarte vorhanden ist auch ein Statusbereich, der Auskunft über den Status der jeweiligen Aktivität gibt. Insbesondere werden hier während der Aktivität registrierte Reagenzien aufgelistet und die unterschiedlichen ID-Listen für exakt diese Aktivität mit Bearbeitungsstatus der Proben angezeigt (F). Da jede Aktivität auf einer eigenen Registerkarte dargestellt wird, ist es durch Blättern möglich, auch die ID-Listen bereits abgearbeiteter Schritte zu jedem beliebigen Zeitpunkt erneut anzusehen. (G) In der Workflow-Definition lassen sich gemäß der Konzeption des GUI für jede Aktivität benötigte Schaltflächen ein- oder ausblenden. Nur tatsächlich benötigte Schaltflächen werden daher auf der Registerkarte der jeweiligen Aktivität angezeigt. Die Schaltfläche ‚Print label' dient dem Ausdrucken von Etiketten, beispielsweise beim Aliquotieren einer Probe oder dem Umfüllen in ein anderes Gefäß. Ein PopUp erlaubt das Eingeben der auszudruckenden ProbenID und die Auswahl der gewünschten Etikettenanzahl. Die Schaltfläche ‚Sample Lost' dient dem Registrieren einer beschädigten Pro-

be unter Verwendung der ‚Lost'-Liste gemäß dem in Abschnitt 5.4.4 konzipierten Proben- und Aliquotenmanagement. Erfolgreich bearbeitete Proben und Aliquoten werden durch die Schaltfläche ‚Sample Done' registriert, ebenfalls gemäß dem in Abschnitt 5.4. 4 konzipierten Proben- und Aliquotenmanagement. Der Status als ‚lost' oder ‚done' registrierter Proben wird im Statusbereich sofort geändert. (H) Eine manuelle Korrektur geänderter Aliquotenanzahlen, wie sie beispielsweise nach Poolen oder Aliquotieren erforderlich ist, wird mit Hilfe der Schaltfläche ‚Set Count' durchgeführt. Sie manipuliert den Inhalt der in Abschnitt 5.4.4 konzipierten ‚Resulting'-Liste. (J) Die vollständige Abarbeitung einer Aktivität wird vom Anwender durch Betätigen der Schaltfläche ‚Completed' bestätigt. Die Workflow-Engine markiert daraufhin die aktuelle Aktivität als abgeschlossen und verändert den Abarbeitungsstatus der Workflow-Instanz. Die in der Abarbeitung folgende Registerkarte wird aktiv, die aktuelle Registerkarte wird inaktiv. (I) Mehrere Schaltflächen stehen für weitere wichtige Funktionen zur Verfügung. Der Verzicht auf einen eigenen Eingabebereich zur Wissensakquise durch den Benutzer wurde in Abschnitt 5.4.6.3 konzipiert. Stattdessen soll ein Dokumentationseditor nach Bedarf ein- und ausgeblendet werden, der durch Abarbeiten der HumanMultiInputActivity in der Workflow-Definition die für den jeweiligen Workflow erforderlichen Datenfelder enthält. Das Ein- und Ausblenden dieses Dokumentationseditors erfolgt durch die Schaltfläche ‚Editor'. Eine Hilfefunktion kann durch die Schaltfläche ‚Help' aufgerufen werden. Häufig benötigte Formeln, beispielsweise zur Bestimmung von Zellzahlen oder zum Berechnen eines notwendigen Volumens von Medium zum Verdünnen einer Zellkonzentration, sind in der ‚ChameleonStation' implementiert und können durch die Schaltfläche ‚Calc' aufgerufen und verwendet werden. Die Schaltflächen ‚Open', ‚Remove Carrier' und ‚Change Carrier' werden zur Interaktion mit den Dockingstationen und den Cryo-Devices benötigt und rufen DCJobs des DCMSFlashRemoteServers auf (‚Open' dient zum Einlesen der Containerdatei, ‚Remove Carrier' zum Zurückschreiben einer aktualisierten Containerdatei aus dem Stationencache und ‚Change Carrier' zum Umkopieren der Containerdatei auf ein anderes Cryo-Device). Die Schaltfläche ‚Report' dient dem automatischen Erstellen eines elektronischen Reports über die bisherige Workflow-Abarbeitung. Zu diesem Zweck wird der Inhalt des Target-Files verwendet und ein Report in .pdf-Format kreiert, das beispielsweise gespeichert oder per Drucker außerhalb des Labors ausgegeben werden kann. Die Schaltfläche ‚Camera' dient dem Aufrufen der Fotofunktion der verwendeten Barcodescanner zum Dokumentieren schwer verbalisierbaren Wissens. Fotos können mit Kommentar, Beschreibung und Informationen zur zugehörigen Probe versehen und gespeichert werden.

5.5.3.4 Die Ansichts- und Exportapplikation

Der Inhalt des Admin-Bereichs einer Trägerplatte oder einer Lagerbox kann mit Hilfe der implementierten Ansichts- und Exportapplikation ‚Content-Viewer' eingesehen werden. Beispielsweise kann der entstandene Inhalt des Target-Files visualisiert werden und gibt Aufschluß über die entstandenen Proben und die Anzahl ihrer Aliquoten. Gleiche Funktion stellt der ‚Content-Viewer' auch hinsichtlich einzelner, auf Lagerboxen organisierter ‚Icebreaker' oder ‚Portlinks' zur Verfügung. Die Selektion eines Flash-Speicherchips und das Auslesen seines Inhalts erfolgt mittels der DCJobs des DCMSFlashRemoteServers. Innerhalb des DCMS nimmt auch der Content-Viewer die Rolle des Commanders ein. Er ermöglicht außerdem den Export des vollständigen Inhalts des User-Areas des selektierten Chips auf ein externes Speichermedium. Nachfolgende Abbildung illustriert das GUI:

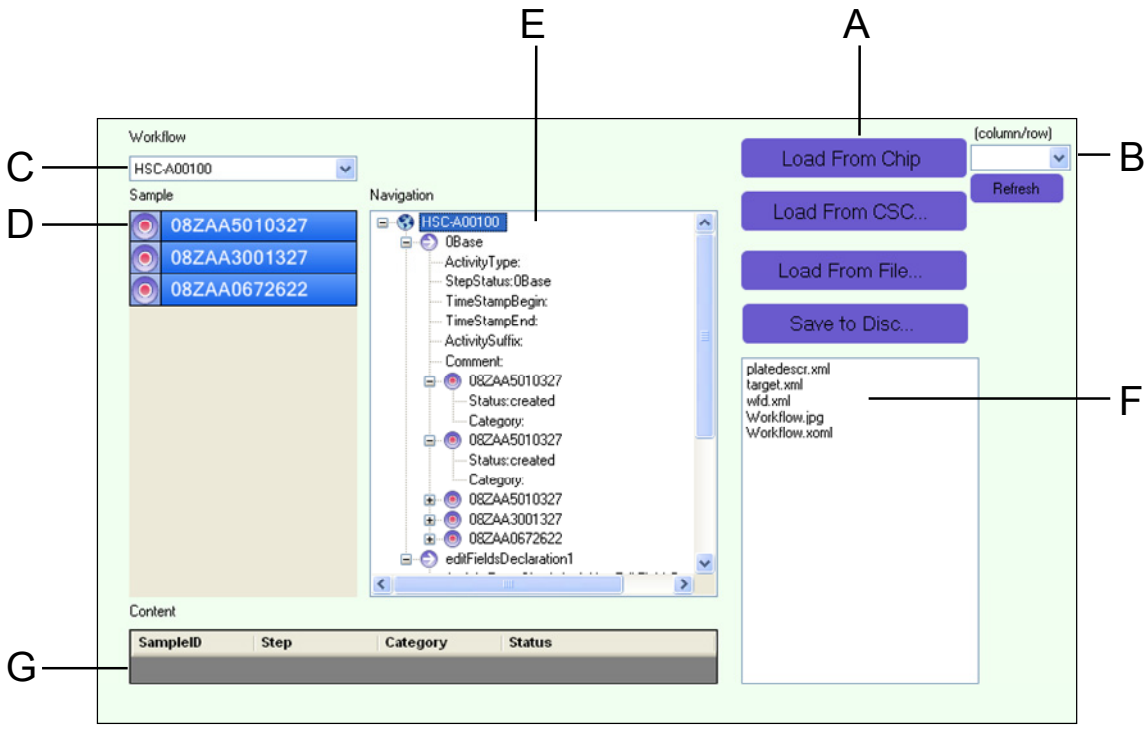

Abbildung 67: Das GUI der Ansichts- und Exportapplikation ‚Content-Viewer'. Diese Applikation erlaubt den selektiven Zugriff auf den Admin-Bereich von Trägerplatten oder Lagerboxen und auf die einzelnen Flash-Speicherchips diskreter Substrate. Der ‚Content-Viewer' ermöglicht unter anderem die Navigation durch das aktuelle Target-File und das lokale Speichern des Inhaltes des User-Areas von Cryo-Devices. (A) Schaltflächen zum Laden und lokalen Speichern des User-Areas von ausgewählten Cryo-Devices. (B) Kombinationsfeld zur Chipauswahl. (C) Anzeige des Namens der in der Containerdatei vorhandenen Workflow-Definition. (D) Übersicht über die im Target-File in der initialen ToDo-Liste registrierten Proben. Die Informationen zu den Proben können durch Anklicken expandiert werden. (E) Navigationsfenster zum Ansehen des Target-Files. (F) Anzeige des Inhalts des User-Areas. Hier sind die Workflow-Definition, die Beschreibung und das Target-File in Form einer Containerdatei enthalten und zusätzlich die Layout-Information des Cryo-Devices. (G) Anzeigefenster für den Bearbeitungsstatus einer ausgewählten Probe.

5.5.4 Interaktion der Systemkomponenten: Online- und Offline-Betrieb

Dieser Abschnitt beschreibt zusammenfassend die Interaktion der implementierten Systemkomponenten anhand nachfolgender schematischer Abbildung. Eine Workflow-Definition kann sowohl ausschließlich in einem einzigen Labor als auch durch Kooperation mehrerer Laboratorien abgearbeitet werden. Hinsichtlich der Verfügbarkeit einer Verbindung zu einer Probendatenbank variiert der Informationsfluß, so dass zwischen einer ‚Online'-Variante und einer ‚Offline'-Variante unterschieden werden kann. Speziell in kollaborativen Szenarien mit Primary Sites ist die Offline-Variante von Bedeutung. Sie gewährleistet die Funktionsfähigkeit des Systems auch bei gestörter oder nicht existierender Verbindung zu einer Probendatenbank, beispielsweise der ‚EurocryoDB'.

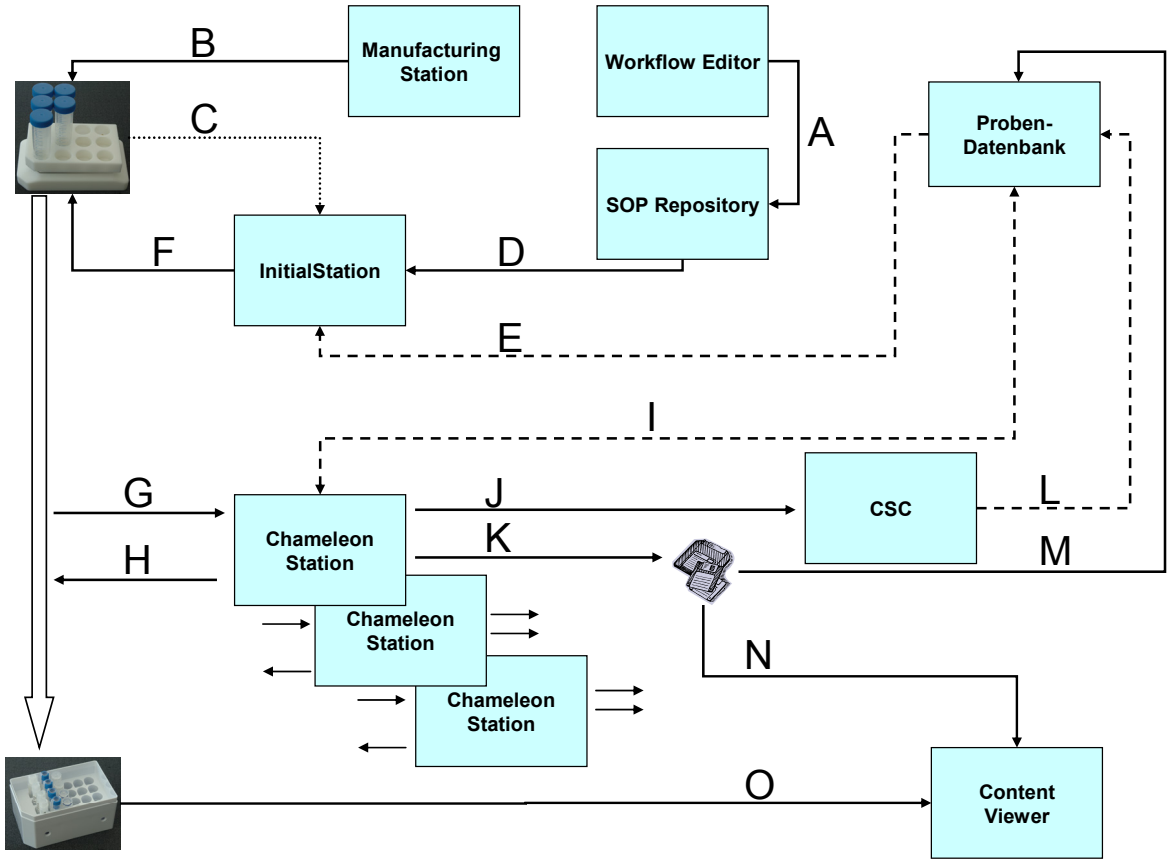

Abbildung 68: Schematische Darstellung der Interaktion der implementierten Systemkomponenten. Je nach Verfügbarkeit einer Verbindung zu einer Probendatenbank und daraus resultierendem Informationsfluß wird zwischen ‚Online'-Variante und ‚Offline'-Variante unterschieden. Bei der ‚Online'-Variante findet Informationsfluß über die gestrichelten Pfeile statt. Bei der ‚Offline'-Variante ist kein Informationsfluß über die gestrichelten Pfeile möglich. (A) Workflow-Definitionen werden mit Hilfe des implementierten Workflow-Editors erstellt. Dazu stehen auch eigene Aktivitäten zur Verfügung, deren Klassencode in einer zusätzlichen Activity-Library organisiert wurde. Durch das Prinzip der Codetrennung kann die Workflow-Definition in Form eines XML-Workflow-Markups erstellt und dezentral in einer Containerdatei gespeichert werden. Unter anderem ist in dieser Datei auch ein Target-File-Template organisiert. Die Containerdateien werden in einem Workflow-Repository bzw. SOP-Repository vorgehalten. (B) Cryo-Devices (beispielsweise Trägerplatte, Lagerboxen und diskrete Substrate) werden mit Hilfe der ‚ChameleonManufacturingStation' formatiert. Ihre Layout-Informationen beeinflussen das GUI und die Funktion der ‚ChameleonInitialStation' (C). (D) Die ‚ChameleonInitialStation' erlaubt die Auswahl einer Workflow-Definition aus dem SOP-Repository und instanziiert die entsprechende Containerdatei. Sie erlaubt das Registrieren zu bearbeitender Proben und erstellt eine initiale ToDo-Liste im Target-File-Template. (E) Bei vorhandener Verbindung zwischen Probendatenbank und ChameleonLab-Installation kann per WebService deklaratives Probenwissen aus der Datenbank abgefragt werden. (F) Die Containerdatei mit modifiziertem Target-File (und eventuelles deklaratives Probenwissen) wird in das User-Area des Cryo-Devices geschrieben. Die Trägerplatte ist nun zur Abarbeitung durch Instanzen der ‚ChameleonStation' bereit. (G) Vor Abarbeitung wird die Containerdatei von der Trägerplatte in den Cache der Laborstation kopiert. Die Workflow-Definition wird an einer oder mehreren Instanzen der ‚ChameleonStation' abgearbeitet. (H) Der Bearbeitungsfortschritt wird auf der Trägerplatte im Target-File dokumentiert und der Status der Workflow-Instanz aktualisiert. Somit kann die Trägerplatte bei Bedarf auch zu anderen Laborstationen transportiert werden und die Bearbeitung dort fortgesetzt werden. Gleiches gilt auch für Laborstationen in kollaborierenden Laboratorien. (I) Bei vorhandener Verbindung zwischen Probendatenbank und abarbeitender ChameleonStation kann das Target-File an die Probendatenbank gesendet werden, um ein Präparationstracking zu ermöglichen.

Aus der Probendatenbank können auch Informationen abgerufen werden, die beispielsweise beim Finalisieren von Proben auf die diskreten Substrate geschrieben werden können. (J) Der Status der Workflow-Instanz und das aktuelle Target-File werden in ein allen Laborstationen gemeinsam zugängliches Verzeichnis (‚CSC') kopiert. (L) Über einen WebService kann der Inhalt des gemeinsamen Verzeichnisses bei Verfügbarkeit der Verbindung zur Probendatenbank zur Probendatenbank übertragen werden. Auf diese Weise ist eine Pufferung von Information möglich, wenn zeitweise keine Verbindung zur Probendatenbank besteht, und die Arbeit an den Laborstationen wird nicht durch fehlende Verbindung zur Datenbank behindert. (K) Bei der Finalisierung diskreter Substrate kann der Inhalt des User-Areas zusätzlich auf externe Speichermedien (beispielsweise USB-Stick) kopiert werden. (M) Die externen Speichermedien können in die Datenbank eingelesen werden, beispielsweise können sie bei einem Transport von Proben von einer Primary Site gemeinsam mit den Proben an das GHRC versendet werden. Selbst wenn dauerhaft keine Verbindung zwischen Primary Site und Probendatenbank bestehen sollte, können die Informationen aus ChameleonLab auf diese Weise in die Datenbank eingepflegt werden. (N) Die Informationen auf den externen Datenträgern können auch im ‚Content-Viewer' angesehen und exportiert werden. (O) Der Content-Viewer kann auch jederzeit eingesetzt werden, um den Inhalt des Cryo-Device und das aktuelle Target-File einzusehen.

5.6 Konzeption von Datenstrukturen für Cryo-Devices und für dezentrale Wissensbewahrung

Im Zusammenhang mit der in Kapitel 4.4 beschriebenen Entwicklung einer elektronischen Lagertank-Infrastruktur ist auch die Konzeption von Datenstrukturen für die verschiedenen ‚Cryo-Devices' erforderlich. Diese Datenstrukturen werden benötigt, um:

- eine elektronische Inventarverwaltung zu realisieren und die Funktionalität der Infrastruktur herzustellen. Gemäß der Beschreibung in Kapitel 4.4 werden in den Speicherchips jedes Cryo-Devices die Anzahl und Position seiner untergeordneten Cryo-Devices gespeichert. Zur Organisation dieser Informationen sind geeignete Datenstrukturen erforderlich.

- die erforderlichen Informationen zur Ansteuerung der untergeordneten Cryo-Devices zu organisieren

- eine elektronische Lokalisierung von Cryo-Devices zu ermöglichen

- Änderungen in der Belegung der Cryo-Devices selbständig zu detektieren und in den Inventarspeichern zu aktualisieren

- Cryo-Devices korrekt bei lesenden und schreibenden Zugriffen anzusteuern und allgemein eine Interaktion zwischen Cryo-Devices zu ermöglichen

- durch kontinuierliches Monitoring der Lagerbedingungen generiertes Wissen auf den Flash-Speicherchips der einzelnen Cryo-Devices abzulegen

Die Datenstrukturen müssen jedoch auch die dezentrale Bewahrung deklarativen Probenwissens und prozeduralen Wissens in Form der in Abschnitt 5.5.1.4 beschriebenen Containerdatei unterstützen. Sie müssen zusätzlich die Interaktion der Systemkomponenten des GHRC-Prototyps mit den relevanten Cryo-Devices (diskrete Substrate, Trägerplatten und Lagerboxen) ermöglichen.

Grundsätzliche Anforderungen an die Funktion der Lagertank-Infrastruktur mit unmittelbaren Auswirkungen auf die Konzeption der Datenstrukturen sind Robustheit und Zuverlässigkeit. Diese Anforderungen ergeben sich aus dem hohen Wert wissenschaftlicher Probensammlungen. Fehlfunktionen, die zu fehlerhafter Information in den Flash-Speicherchips der Komponenten der Infrastruktur führen oder Inkonsistenzen verursachen könnten, müssen vermieden werden. Aus diesem Grund, aber vor allem auch hinsichtlich des Wärmeeintrags in die Lagertanks, wird für den Mikrocontroller der Backplane (siehe Kapitel 4.4) eine effiziente, für die Aufgaben der Backplane optimierte proprietäre Mikrocontrollersoftware entwickelt. Da der Mikrocontroller vorwiegend den Datenfluß steuern soll und aus Gründen der Datensicherheit keine Inhalte von Probendaten oder Workflow-Definitionen verändern darf, wurde bewußt auf höhere Funktionen wie beispielsweise das Parsen von XML-Dateien verzichtet. Der Mikrocontroller greift nur auswertend oder verändernd auf diejenigen Daten zu, die für den Betrieb der Infrastruktur unbedingt erforderlich sind. Einige dieser Daten dürfen auch von der Infrastruktur nicht verändert werden. Aus den geschilderten Anforderungen ergeben sich folgende Bereiche innerhalb der benötigten Datenstrukturen:

- *Protected Area*: für geschützte, Infrastruktur-relevante Daten
- *System Data Area*: für Infrastruktur-relevante Daten, die verändert werden dürfen
- *User-Area*: für Daten, die nicht zum Betrieb der Infrastruktur erforderlich sind, beispielsweise die Workflow-Containerdatei

Hinsichtlich potentiell langer Lagerzeiten im Biobanking und der voraussichtlich langen Nutzungsdauer der elektronischen Lagertank-Infrastruktur und ihrer Cryo-Devices muß es möglich sein, erforderliche Änderungen an den Datenstrukturen durchzuführen, ohne die Kompatibilität mit den Komponenten der Infrastruktur zu beeinträchtigen. Daher bietet sich die Verwendung eines Index an, der die Datenblöcke in den Flash-Speicherchips der Cryo-Devices unter Angabe ihrer absoluten Speicheradresse identifiziert[279]. Ein solcher Index ist auch bei Verwendung von Speicherchips verschiedener Kapazität erforderlich, um den Speicherplatz effizient nutzen zu können. Die Verwendung eines Index ermöglicht zusätzlich eine Optimierung der Datenstrukturen für die verschiedenen Cryo-Devices insofern, als nur die für den jeweiligen Device-Typ relevanten Datenblöcke in der Datenstruktur enthalten sein soll-

[279] Serielle Flash-Speicherchips erlauben unabhängig von Dateisystemen die gezielte Speicherung einzelner Bytes an absoluten Speicheradressen.

ten. Ein typ-spezifischer Index kann für jedes Cryo-Device im Protected Area seines Flash-Speicherchips platziert werden[280]. Seine Startadresse wurde für die unterschiedlichen verwendeten Chiptypen in einem Dictionary abgelegt, der in der Backplane und im DCMSFlashRemoteServer integriert ist. Anhand der JEDEC-Signatur eines Speicherchips wird die Startadresse im Dictionary bestimmt. Mit dieser Methode ist sichergestellt, daß auch für verschiedene Speichertypen der Index gefunden und ausgewertet werden kann.

5.6.1 Cryo-Device-Typen

Der Flash-Speicherchip eines Cryo-Devices soll aus Effizienzgründen nur die jeweils relevanten Datenblöcke enthalten. Gegenwärtig werden folgende Cryo-Device-Typen unterschieden:

- *SingleSample*: ein diskretes Substrat mit eigenem Speicherchip
- *Multisample*: Behälter für mehrere Proben, die keinen eigenen Speicherchip haben, beispielsweise Trägerplatten gemäß Abschnitt 5.4.3.
- *Parent*: Behälter für mehrere Proben mit jeweils eigenem Speicherchip, beispielsweise Lagerboxen gemäß Kapitel 4.3.
- *Wings* (siehe Kapitel 4.4).
- *Racks* (siehe Kapitel 4.4).

Nachfolgende Tabelle ordnet den Cryo-Device-Typen relevante Datenblöcke zu:

Cryo-Device-Typ	**Datenblöcke**
SingleSample	DeviceID, SampleID, Sample Type, Sample Form, Geometry, Anomaly Log, User Files, ,Movement Log, Compressed Temperature Log
Multisample	DeviceID, Geometry, Anomaly Log, User Files, Child List, Movement Log
Parent	DeviceID, Geometry, Anomaly Log, User Files, Child List, Movement Log
Wing	DeviceID, Geometry, Child List, Movement Log, Detailed Temperature Log
Rack	DeviceID, Geometry, Child List, Movement Log, Detailed Temperature Log

Tabelle 13: Auflistung relevanter Datenblöcke für die jeweiligen Cryo-Device-Typen. Gegenwärtig werden SingleSample, Multisample, Parent, Wing und Rack als Cryo-Device-Typen unterschieden. Aus Gründen der effizienten Kapazitätsnutzung serieller Flash-Speicherchips sollen nur benötigte Datenblöcke in den Datenstrukturen der verschiedenen Cryo-Devices vorhanden sein.

[280] Enthält ein Cryo-Device mehrere SPI-Flashspeicher, wird der Index in Chip 0 abgelegt.

5.6.2 Datenblöcke

Die in Tabelle 13 aufgelisteten Datenblöcke werden in diesem Abschnitt näher spezifiziert. Diese Spezifikation beschränkt sich auf eine Übersicht der Datenfelder. Auf detaillierte Angaben zu absoluten Speicheradressen der Datenblöcke innerhalb der Datenstrukturen und auf Angabe des jeweiligen Offsets der Datenfelder innerhalb der Datenblöcke wurde an dieser Stelle bewusst verzichtet.

- Der Typ und die Eigenschaften eines Cryo-Device müssen in der Datenstruktur abgelegt sein. Diesem Zweck dient der nachfolgend beschriebene DeviceID-Datenblock:

Name	**Erläuterung**
Devicetype	Typ des Cryo-Device
JulianDate	Herstellungsdatum
DevMan	Hersteller
ManCode	Hersteller-Chargennummer
Serial	Eindeutige Seriennummer
ProcType	Mikrocontroller-Typ (falls vorhanden)
ProcMan	Hersteller des Mikrocontrollers
ProcManSig	Signatur des Mikrocontroller-Herstellers
ProcSig	Signatur des Mikrocontrollers
SwVersion	Software-Version
NSamPos	Anzahl der Probenpositionen (für Proben ohne eigenen Speicherchip)
nChildPos	Anzahl der Kind-Positionen (mit eigenem Speicherchip; nicht notwendigerweise Proben)
nMemoryChips	Anzahl der Speicherchips des Cryo-Device
JEDECmanufacturer	Speicherchip-Hersteller
JEDECtype	Speicherchip-Typ
JEDECcapacity	Speicherchip-Kapazität

Tabelle 14: Übersicht des DeviceID-Datenblocks und seiner Elemente

- Die im Cryo-Device enthaltenen Proben müssen durch die Datenstruktur identifizierbar sein. Ein entsprechender Datenblock muß die unterschiedlichen ProbenIDs enthalten, die in verschiedenen Biobanken oder kollaborativen Forschungsszenarien benötigt werden. Von weiterem Interesse sind Ort, Datum und Zeitpunkt der Befüllung von Probenröhrchen. Deshalb wurde der nachfolgend beschriebene SampleID-Datenblock entworfen:

Name	**Erläuterung**
Gcount	Länge der GHRC-Proben-ID
GHRC	GHRD-Proben-ID
Tcount	Länge des TRIANTAEXI-Codes
TRIANTAEXI	TRIANTAEXI-Code
Ecount	Länge der Eurocryo-Proben-ID

Eurocryo	Eurocryo-Proben-ID
Tracount	Länge der ID des integrierten Transponders/RFID-tags
transponder	ID des integrierten Transponders/RFID-tags
OriginalInst	Herkunftsinstitut der Probe
OriginalDB	Datenbank-ID der Probe im Herkunftsinstitut
CurrentInst	Gegenwärtiges Institut
CurrentDB	Datenbank-ID der Probe im gegenwärtigen Institut
Sec	Zeitpunkt der Befüllung des Probenröhrchens
Min	Zeitpunkt der Befüllung des Probenröhrchens
Hour	Zeitpunkt der Befüllung des Probenröhrchens
Day	Zeitpunkt der Befüllung des Probenröhrchens
Month	Zeitpunkt der Befüllung des Probenröhrchens
Year	Zeitpunkt der Befüllung des Probenröhrchens
CCount	Länge des Herkunftsland-Kürzels
country	Herkunftsland-Kürzel
LCount	Länge des Herkunftslabor-Kürzels
laboratory	Herkunftslabor-Kürzel

Tabelle 15: Übersicht des SampleID-Datenblocks und seiner Elemente

- Die Art der Probe, wie beispielsweise Vollblut, Plasma oder Viruskultur muß anhand der Datenstruktur erkennbar sein. Ein entsprechender Datenblock könnte eine Zeichenfolge mit biologisch motivierter hierarchischer Struktur enthalten (Beispiel: /PBMC/HIV-negativ):

Name	**Erläuterung**
count	Länge der Zeichenfolge, die die Probenart angibt
Stype	Zeichenfolge, die die Probenart angibt

Tabelle 16: Übersicht des SampleType-Datenblocks und seiner Elemente

- Der physikalische Zustand der jeweiligen Probe muß aus der Datenstruktur hervorgehen. Mögliche Formen sind etwa ‚Suspension' oder ‚Pellet':

Name	**Erläuterung**
count	Länge der Zeichenfolge, die den physikalischen Zustand beschreibt
Sform	Zeichenfolge, die den physikalsichen Zustand einer Probe angibt

Tabelle 17: Übersicht des SampleForm-Datenblocks und seiner Elemente

- Zusätzlich zum physikalischen Zustand einer Probe ist ihr Volumen von Bedeutung, beispielsweise dann, wenn eine bestimmte Präparation ein Mindestvolumen an Probe voraussetzt. Mit Hilfe der Tank-Infrastruktur kann dann eine solche Probe gefunden werden. Für Cryo-Devices mit mehr als einer Probe kann sich der folgende Datenblock entsprechend oft wiederholen:

Name	Erläuterung
Vtype	0 = Volumenkonzept nicht anwendbar 1 = Volumenangabe ist genau 2 = Volumenangabe ist approximiert
Volume	Probenvolumen in Mikrolitern

Tabelle 18: Übersicht des SampleVolume Datenblocks und seiner Elemente

- Die Geometrie eines Cryo-Device-Typs ist von besonderer Bedeutung, unter anderem für die Darstellung des Cryo-Devices und seiner Steckplätze für untergeordnete Cryo-Devices in GUIs. Für jeden Cryo-Device-Typ müssen die geometrischen Daten individuell beschrieben werden:

Name	Erläuterung
DeviceX	Länge des Cryo-Device in X-Richtung in Mikrometern
DeviceY	Länge des Cryo-Device in Y-Richtung in Mikrometern
DeviceZ	Länge des Cryo-Device in Z-Richtung in Mikrometern
ChildNumber	Positionsnummer eines Steckplatzes für untergeordnete Cryo-Devices
ChildX	Position des Steckplatzes in X-Richtung, bezogen auf den Koordinatenursprung des Cryo-Devices in Mikrometern
ChildY	Position des Steckplatzes in Y-Richtung, bezogen auf den Koordinatenursprung des Cryo-Devices in Mikrometern
ChildZ	Position des Steckplatzes in Z-Richtung, bezogen auf den Koordinatenursprung des Cryo-Devices in Mikrometern

Tabelle 19: Übersicht des Geometry-Datenblocks und seiner Elemente

- Das Aufzeichnen von Fehlern oder Abweichungen bei der Lagerung, die Auswirkungen auf die Probe haben können oder bereits hatten, wird benötigt:

Name	Erläuterung
InstName	Name des Instituts
LabName	Labor oder Kryobank, wo die Abweichung aufgetreten ist
MetName	Name der Methode, bei deren Ausführung es zur Abweichung kam (beispielsweise Kryokonservierung)
Errormsg	Fehlermeldung
errorcode	Fehlercode
JulianDate	Datum der Anomalie
Time	Uhrzeit der Anomalie
strings	Durch <CR> getrennte Liste von Bemerkungen

Tabelle 20: Übersicht des Anomaly Log-Datenblocks und seiner Elemente

- Von besonderer Bedeutung hinsichtlich des GHRC-Prototyps ist der User-Area-Datenblock, in dem die Containerdatei abgelegt wird, die unter anderem Workflow-Definition und Target-File enthält. Im User-Area können auch weitere Dateien abgelegt werden, beispielsweise Probendaten in Form einer XML-Datei, die für die elektro-

nische Tankinfrastruktur nicht von Belang sind. Ein virtuelles Dateisystem in Form eines .zip-Archives erleichtert die Identifikation von Inhalten:

Name	Erläuterung
Blocksize	Größe des Datenblocks
data	.zip Archiv für User Files

Tabelle 21: Übersicht des User-Area-Datenblocks und seiner Elemente

- Um für Cryo-Devices zu verwalten, welche untergeordneten Cryo-Devices seine Positionen belegen, kann sich folgender Datenblock entsprechend oft in der Datenstruktur wiederholen:

Name	Erläuterung
ChildNumber	Position des untergeordneten Cryo-Device
ChildDeviceID	Device-ID des untergeordneten Cryo-Device
ChildDtype	Device-Typ des untergeordneten Cryo-Device
SampleID	Liste der Proben-IDs, falls es sich bei den untergeordneten Cryo-Devices um Proben handelt

Tabelle 22: Übersicht des Child List-Datenblocks und seiner Elemente

- Um Transportvorgänge oder den Wechsel von Lagerpositionen von Proben oder Cryo-Devices innerhalb des Lagersystems zu dokumentieren, wurde folgender Datenblock konzipiert:

Name	Erläuterung
ParentType	Typ des übergeordneten Cryo-Device
Position	Position im übergeordneten Cryo-Device
ParentSerial	Seriennummer des übergeordneten Cryo-Device
Institute	Biobank-ID
Tank	ID des Kryo-Lagertanks
JulianDate	Datum der Einlagerung in das übergeordnete Cryo-Device

Tabelle 23: Übersicht des Movement Log-Datenblocks und seiner Elemente

- Während der Lagerung von Proben sollen auf ihre Speicherchips für jeden Tag die durchschnittlichen, maximalen und minimalen Temperaturwerte geschrieben werden:

Name	Erläuterung
JulianDate	Datum des aufgezeichneten Tages; weitere Aufzeichnungen im Abstand von jeweils einem Tag
mean	Durchschnittstemperatur
min	Gemessene Minimaltemperatur
max	Gemessene Maximaltemperatur

Tabelle 24: Übersicht des Compressed Temperature Log-Datenblocks und seiner Elemente

- Lagerboxen und Wings sollen Temperaturkurven aufzeichnen. Messungen sollen in definierten Zeitintervallen erfolgen, und jeder dieser Meßwerte soll in der Datenstruktur dokumentiert werden:

Name	Erläuterung
JulianDate	Datum des aufgezeichneten Tages
Time	Zeitpunkt der Messung; Intervall nach Definition
temperature	Gemessener Temperaturwert

Tabelle 25: Übersicht des Detailed Temperature Log-Datenblocks und seiner Elemente

Einige der konzipierten Datenblöcke dürfen weder von der elektronischen Tank-Infrastruktur noch von den Systemkomponenten des GHRC-Prototyps verändert werden. Sie müssen daher im geschützten Speicherbereich abgelegt werden. Es handelt es sich um:

- DeviceID
- SampleID
- SampleType
- Sample Form
- Geometry

Andere Datenblöcke hingegen müssen unbedingt im ungeschützten Speicherbereich abgelegt werden, da sie von der Tank-Infrastruktur ergänzt oder von den Systemkomponenten des GHRC-Prototyps bearbeitet werden müssen. Es handelt sich um:

- User-Area
- Compressed Temperature Log
- Detailed temperature Log
- Movement Log

Die gemäß Tabelle 13 aus den in diesem Abschnitt beschriebenen Datenblöcken zusammengesetzten Datenstrukturen stehen zur Formatierung der unterschiedlichen Cryo-Devices mit Hilfe der in Abschnitt 5.5.3.1 beschriebenen ChameleonManufacturingStation zur Verfügung. Die Formatierung von diskreten Substraten, Multisamples (Trägerplatten) und Parents (Lagerboxen) erfolgt unter Verwendung der Dockingstationen und der DCUnit DCMSFlash-RemoteServer.

6 Diskussion

Wesentlicher operativer Aspekt im Biobanking ist die vitale Langzeitlagerung von Zellen mit dem Ziel, ihre strukturelle Integrität und die zelltyp-spezifische Funktionsfähigkeit zu erhalten und somit eine hohe Probenqualität zu bewahren. In diesem Zusammenhang können Beschädigungen von Zellmembran, von Zellorganellen oder des Zytoskeletts als Beispiele struktureller Schädigungen genannt werden, die aus intra- oder extrazellulärer Eisbildung oder aus osmotischen Effekten bei der Kryokonservierung auftreten können. Eine verminderte zelluläre Immunantwort mononukleärer Immunzellen, eine geringere Produktion von zelltyp-spezifischen Substanzen wie beispielsweise Insulin im Zusammenhang mit Langerhans'schen Inseln aus dem Pankreas oder allgemein ein verminderter Metabolismus können als Beispiele eingeschränkter zellulärer Funktionsfähigkeit angeführt werden. Solche Effekte können unter anderem durch Waschen von Zellen [Disis 2006], durch Transport [Betensky 2000] oder durch Kryokonservierung [Weinberg 2000] entstehen und sind damit Folge von Handhabungs- oder Bearbeitungsvorgängen.

Der Wert wissenschaftlicher Probensammlungen ergibt sich unmittelbar aus der erreichten Probenqualität, die ihrerseits von Handhabung, Kryokonservierung und Bearbeitung abhängt; er ist jedoch auch unmittelbar abhängig von der Qualität zugehörigen Probenwissens und dessen Management, das nach nach dem Prozeßmodell von Reinmann-Rothmeier (vgl. [Reinmann-Rothmeier 2000]) die Aspekte der Wissensrepräsentation, Wissenskommunikation, Wissensgenerierung und Wissensnutzung umfaßt[281].

Das im Kontext von Biobanking vorkommende Wissen konnte anhand der Unterscheidung von Wissenstypen gemäß [Ryle 1949] und [Oberauer 1993] klassifiziert werden in deklaratives und prozedurales Probenwissen. Wissen, das unabhängig von einer Probenbearbeitung zur Verfügung steht oder erhoben werden kann, beispielsweise Spenderdaten, Probenherkunft, Anamnese oder Datum und Zeit der Probenentnahme, ist deklaratives Wissen. Wissen, das zur Lagerung, Bearbeitung oder Handhabung einer Probe erforderlich ist, stellt prozedurales Wissen dar. Wissen, das durch Lagerung, Bearbeitung oder Probenverwendung entsteht, beispielsweise Labordokumentation, Lagerdaten oder festgestellte Probeneigenschaften, kann wiederum als deklaratives Wissen klassifiziert werden, das durch Wissensnutzung entsteht.

Das prozedurale Wissen im Biobanking und in biomedizinischen Forschungslaboratorien wird in Form zelltyp-spezifischer ‚Protokolle' repräsentiert. Protokolle wurden aus Sicht verschiedener wissenschaftlicher Disziplinen betrachtet. Dabei hat sich aus Sicht des Wis-

[281] In [Koch 1999] und [Probst 1998] werden Wissensidentifikation, Wissenserwerb, Wissensentwicklung, Wissensverteilung, Wissensnutzung und Wissensbewahrung als Bausteine des Wissensmanagements genannt.

sensmanagements gezeigt, daß ihre iterative Optimierung und ihre Anwendung innerhalb einer wissenschaftlichen Gemeinschaft anhand des SECI-Modells und der Spirale des Wissens ([Nonaka 1997], [Nonaka 2000]) beschrieben werden kann. Aus wirtschaftsinformatischer Sicht wurde deutlich, daß es sich bei biomedizinischen Protokollen faktisch um Workflow-Definitionen handelt, deren einzelne Protokollschritte den einzelnen Aktivitäten von Workflows entsprechen. Aus graphentheoretischer Sicht wurde deutlich, daß ein Protokoll einem gerichteten Graphen entspricht und die einzelnen Protokollschritte den Kanten des Graphen. Je nach Eingangs- und Ausgangsgrad stellen die Knoten entweder die ursprüngliche Probe, Probenzwischenprodukte oder Probenprodukte dar. Durch geeignete Knoten- und Kantenmarkierungen kann ein Proben- und Aliquotenmanagement repräsentiert werden.

Die Untersuchung von Protokollen hat aber auch gezeigt, daß Protokolle keine definierte Detailschärfe besitzen: während manche Protokolle sehr detailliert formuliert sind und das erforderliche prozedurale Wissen im Detail beschreiben, erfordern andere Protokolle die Substitution lückenhafter Prozedurbeschreibungen durch individuelles Erfahrungswissen und den Übergang in einen erfahrungsgeleiteten Handlungsmodus gemäß [Büssing 2002]. Erfahrungsgeleitetes Arbeitshandeln erfolgt auf Basis impliziten Wissens [Carus 1996], das durch Routinisierung [Anderson 1992] bzw. Prozeduralisierung ([Berry 1987], [Büssing 1999]) entstanden ist. Die Problematik dabei ist, daß implizites Wissen fehlerhaft sein kann, ohne als fehlerhaft erkannt zu werden [Herbig 2001b], und daß sich gemäß [Speelman 1998] aus Unterschieden im individuellen impliziten Wissen ([Polanyi 1966], [Polanyi 1985]) Unterschiede im Arbeitshandeln ergeben können, obwohl ‚per se' identische Protokolle ausgeführt werden. Konsequenz daraus ist, daß bei der Probenbearbeitung durch verschiedene Personen Nuancen oder größere Abweichungen auftreten können, die eine Vergleichbarkeit von Ergebnissen beispielsweise im Zusammenhang kollaborativer Impfstoffstudien behindern, die gemeinsame Wissensbasis einer ‚Learning Community' [Mandl 2000] verfälschen oder auch eine detaillierte Reproduzierbarkeit einer Probenpräparation unmöglich machen können. Im Zusammenhang mit der Videoanalyse einer Probenpräparation, bei der die einzelnen Schritte eines Protokolls mit ihrer tatsächlichen Ausführung verglichen wurden, wurde ein solcher Übergang in einen erfahrungsgeleiteten Handlungsmodus deutlich: es wurden Handlungen und Details beobachtet, die im Protokoll nicht beschrieben waren. Ein ähnlicher Einfluß erfahrungsgeleiteten Arbeitshandelns und individueller Prioritäten auf entstehende Labordokumentation konnte ebenfalls beobachtet werden. Es wurde deutlich, daß manche Schritte sehr ausführlich dokumentiert wurden, während andere in der Dokumentation nicht erwähnt wurden.

Das übliche Management von Protokollen in biomedizinischen Forschungslaboratorien wurde untersucht. Dabei wurden verschiedene Probleme deutlich, unter anderem das Risiko von Verwechslungen, das aus referenzierender Zuordnungsmethodik resultiert. Dieses Risiko liegt sowohl bei papiergebundenen Protokollen wie auch bei handschriftlicher Akquise von Labordokumentation vor und kann im Extremfall zu Personengefährdung führen wie auch ge-

nerell kollaborative Forschung korrumpieren. Eine weitere Schwierigkeit ist häufig die Ko-Existenz verschiedener Versionen eines Protokolls, die zu Unterschieden bei der Probenbearbeitung führen kann.

Herkömmliche Ansätze, um solche Unterschiede bei der Protokollausführung und Wissensakquise in biomedizinischen Forschungslaboratorien zu reduzieren und damit die Qualität der Probenbearbeitung und der Labordokumentation zu verbessern, wurden untersucht. Sie basieren im wesentlichen auf der Fokussierung auf jeweils eine verbindliche Version eines Protokolls (SOP) und eines verbindlichen Dokumentationsformulars, die mit Hilfe von EDMS erstellt und verwaltet werden können. Hinsichtlich der Problematik handschriftlicher Dokumentation wurde das Prinzip elektronischer Laborbücher an einem Beispiel untersucht [Fromme 2000]. Es wurde deutlich, daß die Verwendung elektronischer Laborbücher die Wissensakquise, Wissensbewahrung und Wissenskommunikation unterstützen kann. Jedoch können EDMS und elektronische Laborbücher die Kernprobleme in biomedizinischen Laboratorien nicht beseitigen, nämlich (1) den Einfluß impliziten Wissens bei der Ausführung von Protokollen, (2) den Einfluß individueller Sorgfalt bei manueller Datenerfassung und (3) eine reale Gefahr von Fehlzuordnungen zwischen Probe, zugehörigem Protokoll und entstandener Labordokumentation durch referenzierende Zuordnung.

Als ein weiterer Ansatz zur Minimierung von Unterschieden bei der Probenbearbeitung wurde klassische Laborautomatisierung nach [Mahaffey 1991] eingehend untersucht. Die Intention ihrer Entwicklung ist die Vermeidung individueller Einflüsse und resultiert in automatisierter, rein maschineller Probenbearbeitung. Sowohl vollständige Laborautomatisierung (TLA) [Sasaki 1998] wie auch modulare Laborautomatisierung in Form modularer Workcells [Felder 1998] entsprechen nicht den identifizierten Anforderungen biomedizinischer Forschungslaboratorien, die im wesentlichen in hoher Flexibilität und Kompatibilität mit einer Vielzahl verschiedener Protokolle mit einem hohen Anteil manueller Arbeitsschritte bestehen. Sinngemäßes gilt für die zu Laborautomatisierungssystemen komplementären Laborinformations-Managementsysteme (LIMS), die zur Akquise und Aufbereitung bei automatisierter Bearbeitung entstehenden Wissens eingesetzt werden. Auch bei Laborautomatisierung ist die Problematik der Verwechselung durch zentrale Datenhaltung und referenzierende Zuordnung gegeben.

Es hat sich gezeigt, daß in biomedizinischen Forschungslaboratorien eine andere Form des Managements deklarativen und prozeduralen Probenwissens benötigt wird. Eine Möglichkeit zur Verbesserung der Wissensbewahrung und Wissensverteilung ist ein Ansatz des Fraunhofer-IBMT, der in einer mechanischen Kopplung von kryotolerantem elektronischem Speichermedium und Probe besteht und auf eine zusätzliche dezentrale Datenhaltung direkt bei der Probe abzielt. Diese Ergänzung zur konventionellen Datenhaltung in Probendatenbanken soll fehlerhafte Zuordnungen zwischen Proben und zugehörigem Wissen ausschließen

und eine zuverlässige Probenidentifizierung gewährleisten, insbesondere vor dem Hintergrund potentiell langer Lagerzeiten in Biobanken und beim Probenaustausch mit anderen Institutionen. Dieser Ansatz fokussierte ursprünglich ausschließlich auf die Bewahrung und Verteilung existierenden deklarativen Probenwissens, nicht aber auf die Sicherstellung seiner Qualität. Diese ist, wie oben dargelegt, abhängig von der Wissensgenerierung und der Wissensakquise bei der Probenbearbeitung[282], deren Güte wie beschrieben unmittelbaren Einfluß auf die Qualität der Probe hat. Um einen hohen wissenschaftlichen Wert von Probensammlungen zu erreichen, ist eine qualitativ hochwertige Bewahrung deklarativen Probenwissens alleine somit nicht hinreichend. Vielmehr muß eine Verwechslungssicherheit auch hinsichtlich Protokollen und Labordokumentation erreicht werden. Zusätzlich ist eine Methode erforderlich, um eine einheitliche Ausführung von Protokollen zu erreichen.

Der eigene Ansatz der vorliegenden Arbeit baut auf dem Ansatz des IBMT zur dezentralen Wissensbewahrung bei der Probe auf. Basierend auf der Analogie zwischen biomedizinischen Protokollen und wirtschaftsinformatischen Workflows, den festgestellten Anforderungen an Wissensaustausch und Wissensbewahrung und hinsichtlich der erforderlichen Flexibilität und Kompatibilität biomedizinischer Forschungslaboratorien mit einer Vielzahl von Protokollen wurde ein geeignetes CAQ-System wie folgt charakterisiert: (1) Kopplung von Probe und Protokoll, (2) Probe steuert ihre eigene Präparation und (3) Kopplung von Probe und Labordokumentation. Das prozedurale Wissen für eine Probenbearbeitung soll in Form einer elektronischen Workflow-Definition auf dem Speicherchip gespeichert sein, der zur dezentralen Wissensbewahrung mechanisch mit der Probe verbunden ist. Dadurch wird eine Verwechselungssicherheit erreicht und Fehlpräparationen werden vermieden. Diese Workflow-Definition soll durch ein Workflow-Managementsystem (WfMS) abgearbeitet werden, das Laborpersonal schrittweise durch die Präparation der Probe führt und dabei generiertes Proben- und Prozeßwissen detailliert und einheitlich auf dem Speicherchip akquiriert. Durch die Benutzerführung unter Bereitstellung detaillierten prozeduralen Wissens soll der Einfluß individuellen impliziten Wissens reduziert, eine Vergleichbarkeit von Präparationen erreicht und die Qualität von Proben und entstandenem Wissen gesichert werden. Zusätzlich sollen Laborgeräte aus der Workflow-Definition heraus gesteuert werden können.

Um die grundsätzliche Machbarkeit des eigenen Ansatzes zu evaluieren, wurden Anforderungen an ein erstes Testsystem analysiert und abstrahiert. Zu diesem Zweck wurden repräsentative Protokolle betrachtet und ihre Eigenschaften extrahiert. Es wurde festgestellt, daß biomedizinische Laborprotokolle eine Vielzahl manueller Bearbeitungsschritte enthalten, daß zusätzlich semimanuelle Bearbeitungsschritte (‚Benutzer bedient Gerät') vorkommen, daß manuelle Transportvorgänge erforderlich sind, daß Präparationsschritte rein sequentiell abge-

[282] Deklaratives Probenwissen entsteht zum großen Teil durch die Bearbeitung von Proben. Seine Korrelierbarkeit mit Probeneigenschaften wird durch die Qualität der Wissensakquise beeinflusst.

arbeitet werden und daß keine Kontrollstrukturen in den Protokollen vorkommen. Es wurde auch deutlich, daß benötigte Geräte je nach Protokoll in anderer Reihenfolge benutzt werden. Eine einheitliche Prozeßstrecke kann daher nicht realisiert werden. Stattdessen muß eine flexible Gerätenutzung und möglich sein. Aus der Erwartung unterschiedlichster Proben folgte ferner die Anforderung der Kompatibilität mit entsprechend vielen Protokollen.

Es wurde evaluiert, ob die festgestellten Anforderungen auf Basis einer existierenden Laborautomatisierungssoftware mit zentraler Architektur umgesetzt werden können. Bestandteile der Software waren ein System Controller, Geräteapplikationen und eine zentrale Datenbank zur Definition der Workflows (‚Runs') in proprietärem Format. Dabei zeigte sich, daß der System Controller (ein event-basierter Scheduler) verwendet werden konnte, und daß die Struktur der zugehörigen Run-Datenbank der Anforderung gezielter Wissensverteilung entsprach. Es zeigte sich auch, daß die Kommunikationsschnittstellen des System Controllers verwendet werden konnten, um eine Interaktion mit Laborpersonal zu etablieren. Dazu mußte eine zusätzliche Applikation konzipiert werden, deren Verhalten durch Abarbeitung einer Rundefinition gesteuert werden konnte, um auf Basis eines ebenfalls konzipierten Kommandosatzes die Ausgabe von explizitem prozeduralem Wissen und die Wissensakquise mit Hilfe vordefinierter Formulare zu ermöglichen. Die event-basierte Konzeption des Schedulers stellte sich als nützlich heraus, um auch zeitlich schwer kalkulierbare Bearbeitungsschritte zu integrieren und um eine synchronisierte Verteilung der Bearbeitungsschritte auf verschiedene Instanzen der Applikation im Labor zu realisieren. Mit Hilfe dieser Umsetzung wurde es möglich, Personen als Aktoren in ein High-Throughput-fähiges Automatisierungssystem zu integrieren. Als wesentliche Limitierungen des Testsystems wurden die ausschließliche Speicherung der Workflow-Definitionen in einer zentralen Datenbank, ihre proprietäre Struktur, die Verwendung assoziierter Dateien und vordefinierter Formulare und die ausschließliche Nutzbarkeit des Systems für ein einziges Protokoll pro Zeitpunkt identifiziert.

Das Testsystem diente in der Folge zur Identifikation von Anforderungen an den GHRC-Prototypen. Eine solche Anforderung ist die Verwendung einer Workflow-Beschreibungssprache, die deutlich über den Befehlssatz des Testsystems hinausgeht, um auch komplexe Protokolle im Rahmen des GHRC-Projektes repräsentieren zu können, die mehrere Präparationspfade enthalten oder Kontrollstrukturen erfordern. Auch die veränderte Architektur aus autonomen Laborstationen ohne zentralen System Controller, aber mit jeweils integrierter Workflow-Engine, die Konzeption des GUI auf Basis einer Ergonomieuntersuchung nach DIN EN ISO 9241 und die Workflow-Vorschaufunktion ergaben sich aus der Evaluierung des Testsystems und aus dabei identifizierten Anforderungen. Weitere wesentliche Anforderungen an den GHRC-Prototypen ergaben sich auch aus den Aspekten kollaborativer Forschungsszenarien. Eine besondere Rolle spielt die globale, verteilte Protokollausführung. Die Notwendigkeit von Unterbrechbarkeit und Fortsetzbarkeit eines Workflows in einem anderen Labor oder an anderen Stationen wurde deutlich. Zusätzlich zeigte sich, daß eine Mög-

lichkeit benötigt wird, Workflow-Definitionen ohne Änderungen zwischen Laboratorien auszutauschen. Im Gegensatz zum Testsystem wurde nun die vollständige dezentrale Speicherung der Workflow-Definition auf dem Speicherchip des Probenröhrchens erforderlich, ebenso die Erfassung von Labordokumentation ohne vordefinierte assoziierte Eingabeformulare. Hinsichtlich der Integration automatisierter Aktivitäten in die Workflow-Definitionen zeigte sich hier die Notwendigkeit, abstrahierte Meta-Aktivitäten zu verwenden, die gemeinsame Befehle und Parameter jeweils einer Geräteklasse beherrschen. Die erforderliche Ausführung der Meta-Befehle und ihrer Parameter durch konkrete Laborgeräte verdeutlichte die Notwendigkeit, eine modulare Systemarchitektur zu verwenden, an die wiederverwendbare, modulare Softwarekomponenten eines geeigneten Device Integration Frameworks angebunden werden können. Eine Untersuchung impliziter Anforderungen anhand einer Videoanalyse zeigte unter anderem die Notwendigkeit, eine umfangreiche Aliquotenverwaltung zu realisieren, die den Abarbeitungsstatus jedes Probenröhrchens mit Zeitstempel individuell erfaßt und dokumentiert. Dies stellt die Basis dar, um künftig biologische Effekte mit zeitlichen Aspekten korrelieren zu können, insbesondere für solche Proben, deren Qualität von der Einhaltung enger zeitlicher Rahmenbedingungen bei der Bearbeitung abhängig ist. Eine Dokumentation aller in einem Versuch vorhandenen Proben wurde mittlerweile auch für Elab 2.0 [Schröder 2008] angekündigt.

Ähnlich wie die Anforderungen an das System, haben sich mit der technologischen Weiterentwicklung von Substraten und Lagertank-Infrastruktur auch die Anforderungen an Datenstrukturen für die dezentrale Wissensbewahrung auf den Speicherchips der Proben verändert. Anforderung an die erste Datenstruktur waren lediglich Langzeitkompatibilität und Erweiterbarkeit. Diese konnten durch Verwendung von XML-Strukturen realisiert werden. Es war damit unter anderem möglich, sowohl eine dezentrale Referenz auf die Rundatenbank des Testsystems zu definieren wie auch Strukturen für die dezentrale Speicherung von Labordokumentation. Im Zusammenhang mit dem GHRC-Prototypen und speziell mit der Realisierung der elektronischen Lagertank-Infrastruktur jedoch wurde es notwendig, auch die Daten für die Interaktion von Cryo-Devices und Infrastruktur dezentral zu speichern. Es wurde analysiert, welche Datenstrukturen für den Betrieb des jeweiligen Cryo-Device in Abhängigkeit seiner Organisationsstruktur erforderlich sind. Dies hat die Konzeption einer RAW-Datenstruktur motiviert, die mit der Infrastruktur und ihrem Betriebssystem kompatibel ist. Die RAW-Struktur erlaubt auch die Integration beliebiger Dateien in einem besonderen Speicherbereich (User Area). Dazu zählen unter anderem die für den Betrieb des GHRC-Prototypen benötigten Dateien, beispielsweise eine Containerdatei mit XML-Workflow-Markup und Target File. Die elektronische Lagertank-Infrastruktur kann aus Sicherheitsgründen XML-Dateien im User Area nicht interpretieren oder verändern.

Durch Konzeption und Implementierung der an den GHRC-Prototyp identifizierten Anforderungen und durch Umsetzung der konzipierten Datenstruktur für Cryo-Devices wurde

die vollständige Umsetzung des eigenen Ansatzes erreicht. Entstanden ist ein skalierbares CAQ-System für biomedizinische Forschungslaboratorien, das dezentrales Wissens- und Workflow-Management miteinander vereint [Durst 2008]. Das System kann durch schrittweises Bereitstellen des benötigten expliziten prozeduralen Wissens den Einfluß individuellen, erfahrungsgeleiteten Arbeitshandelns vermindern[283] und die Qualität der Probenbearbeitung standardisieren. Die automatisierte Akquise von Prozeßwissen in Verbindung mit Zeitstempeln und die Wissensakquise anhand vordefinierter Datenfelder minimiert den Einfluß individueller Prioritäten und führt zu vergleichbarer, reproduzierbarer Labordokumentation. Die Speicherung[284] von Workflow-Definition und Akquise von Labordokumentation bei der Probe erhöht die Qualität der Wissensbewahrung, ermöglicht einen vereinfachten Proben- und Wissensaustausch und erleichtert die Zusammenarbeit von Laboratorien. Organisationales Lernen und das Etablieren neuer Workflows wie auch die Wissensnutzung allgemein werden erleichtert. Das System kann als prinzipieller Machbarkeitsbeweis für kollaborative Zusammenarbeit von Laboratorien dienen und wird bereits in den GHRC-Laboratorien in Sulzbach und in der GHRC-Primary Site am Tygerberg Campus der Stellenbosch University (Kapstadt, Südafrika) für die Aufarbeitung HIV-infizierter Blutproben verwendet [Ihmig 2009]. Seitens führender LIMS-Hersteller wurde ein Interesse artikuliert, ChameleonLab an ihre Systeme anzubinden.

---

[283] Dennoch ist implizites Begriffswissen unabdingbar, um Handlungsanweisungen zu verstehen.

[284] Das System leistet ein dezentrales Management deklarativen und prozeduralen wie auch aus der Nutzung prozeduralen Wissens entstandenen deklarativen Wissens. Alle drei im Biobanking identifizierten Wissensarten werden durch den GHRC-Prototypen in den Speicherchips der Cryo-Devices gespeichert.

7 Ausblick

- Gegenwärtig beträgt die Kapazität des Flash-Speicherchips in den Röhrchen 1 MB. Durch höhere Speicherkapazität könnte die Workflow-Definition beispielsweise um Video-Dateien ergänzt werden, die schwierig zu verbalisierendes Wissen (beispielsweise Pipettiergeschwindigkeit bei Überschichten oder Schräghaltung des Röhrchens) illustrieren. Dadurch kann ein implizites Lernen durch Beobachtung erfolgen, das der Sozialisation im SECI-Modell entspricht und eine Meister-Lehrlings-Beziehung imitiert.
- Bei höherer Speicherkapazität wäre es auch möglich, die vollständige Präparation per Video zu dokumentieren und zusätzlich zur vom System erstellten Labordokumentation auf dem Chip zu speichern. Auch wäre es möglich, eine zusätzliche gesprochene Dokumentation bei der Probe zu hinterlegen. Dies würde eine vollständige Reproduzierbarkeit gewährleisten.
- Im Zusammenhang mit der Implementierung von Service-orientierten Architekturen (SOAs) könnte das System über auf dem Chip definierte zu beanspruchende Dienste jeweils auf gültige rechtliche Limitierungen o.ä. zugreifen.
- Durch Integration eines Microcontrollers könnten Proben miteinander unter Verwendung von Netzwerken kommunizieren, Daten austauschen, eine Bearbeitungsreihenfolge definieren oder die jeweils aktuelle Version eines Workflows herunterladen.
- Durch integrierte Sensorik könnte ein Probenröhrchen rechtzeitig kritische Zustände der enthaltenen Probe melden. Dies wäre im Zusammenhang mit der Einhaltung zeitlicher Constraints bedeutsam. Zusätzlich könnte eine Plausibilitätsprüfung des Inhalts anhand optischer Sensorik erfolgen (Lichtbrechung, Farbe, usw.).
- Das System könnte über eine Anbindung von Sensorik in Kleidung die Arbeitsfähigkeit des Users einschätzen und Rechte einschränken oder zuteilen, um Proben zu schützen.
- Aktuell wird die Integration der Kryo-Werkbank und des Entnahmeturms in den GHRC-Prototypen durchgeführt. Eine Erweiterung des Systems um weitere Laborgeräte kann durch Implementierung geeigneter DCUnit-Module und um Erweiterung der Activity Library um geeignete Meta-Aktivitäten und Parameter erfolgen.
- Durch bidirektionale Meta-Aktivitäten wäre es möglich, die Ausführung einer Menge von automatisierten Workflow-Schritten ergebnisbasiert zu beeinflussen, beispielsweise einen Regelkreis aus Zellkonzentrationsmessung und Suspensionsaktivität.
- Anhand generierter Reports könnten Prozeßkostenrechnungen aufgestellt und Workflows zusätzlich hinsichtlich verschiedener Zielfunktionen optimiert werden.

8 Zusammenfassung

Ausgangspunkt und Kontext der vorliegenden Arbeit waren Bestrebungen des Fraunhofer-IBMT, die Qualität der Wissensbewahrung im Biobanking durch mechanische Kopplung von kryotolerantem elektronischem Speichermedium und Probe zu verbessern. Dieses Konzept fokussierte auf die Bewahrung existierenden deklarativen Probenwissens, nicht aber auf die Sicherstellung seiner Qualität. Diese ist abhängig von der Wissensgenerierung und der Wissensakquise bei der Probenbearbeitung, deren Güte unmittelbaren Einfluß auf die Qualität der Probe hat. Die Bearbeitung von Proben ist neben ihrer Langzeitlagerung einer der operativen Aspekte im Biobanking. Sie erfolgt anhand spezifischer Methoden, die in Form von Protokollen vorliegen. Protokolle enthalten prozedurales Wissen, jedoch ist ihre Detailschärfe undefiniert. Der in biomedizinischen Forschungslaboratorien übliche Umgang mit Protokollen wurde in der vorliegenden Arbeit untersucht. Dabei wurden verschiedene Probleme deutlich, unter anderem Risiken aus referenzierender Zuordnungsmethodik, erfahrungsgeleitetes Arbeitshandeln zur Substitution lückenhafter Protokolle, daraus resultierende Unterschiede bei der Probenbearbeitung und Unterschiede bei der handschriftlichen Akquise von Labordokumentation. Bestrebungen, um Unterschiede bei der Protokollausführung und Wissensakquise in biomedizinischen Forschungslaboratorien zu reduzieren, wurden evaluiert, unter anderem EDMS und elektronischer Laborbücher. Ein weiterer Ansatz, um die Unterschiede bei der Probenbearbeitung zu minimieren und die Qualität von Proben zu erhöhen, sind Laborautomatisierungssysteme zur automatisierten Probenbearbeitung. Sie wurden zusammen mit den komplementären LIMS, die der Akquise und Aufbereitung entstehenden Wissens dienen, vor dem Hintergrund der Anforderungen biomedizinischer Forschungslaboratorien evaluiert. Aus informatischer Sicht von besonderem Interesse waren Aspekte der Laborautomatisierungssoftware. In diesem Zusammenhang wurden System Controller und Ansätze zur Integration von Laborgeräten in Automatisierungssysteme betrachtet. Die erforderliche Flexibilität in biomedizinischen Laboratorien und die Vielzahl verschiedener Protokolle mit einem hohen Anteil manueller Arbeitsschritte weichen jedoch deutlich vom Einsatzspektrum klassischer Laborautomatisierung ab und machen die Verwendung solcher Systeme unmöglich.

Es erfolgte eine Einordnung biomedizinischer Protokolle aus Sicht verschiedener wissenschaftlicher Disziplinen. Auf Basis dieser Einordnung und unter Einbeziehung der technologischen Entwicklungen des Fraunhofer-IBMT wurde der eigene Ansatz der vorliegenden Arbeit formuliert.

Es wurden Konzeption und Implementierung eines ersten Testsystems beschrieben, das dazu diente, die Machbarkeit des eigenen Ansatzes zu evaluieren und Anforderungen an Folgesysteme zu identifizieren. Das Testsystem wurde auf Basis existierender Laborautomatisie-

rungssoftware aufgebaut. Als System Controller kam ein event-basierter Scheduler zum Einsatz, um prozedurales Wissen gezielt an Laborpersonal zu verteilen und den Präparationsablauf zu koordinieren. Dies erfolgte mittels mehrerer Instanzen einer Applikation zur Interaktion mit dem Laborpersonal. Das Testsystem wurde unter Verwendung der Schnittstellen und der Workflow-Struktur des Schedulers aufgebaut. Eine erste Datenstruktur für dezentrale Wissensbewahrung wurde konzipiert vor dem Hintergrund langer Lagerzeiten und erforderlicher Flexibilität. Sie diente der dezentralen Bewahrung deklarativen Wissens und konnte vom Testsystem generierte Labordokumentation aufnehmen. Die Steuerung des Testsystems durch die Probe konnte simuliert werden. Eine erste Integration der Datenstruktur in prototypische Lagertechnologie wurde durchgeführt.

Funktionale Limitierungen des Testsystems und Limitierungen aus Anwendersicht wurden analysiert und abstrahiert. Aus diesen Limitierungen des Testsystems konnten Anforderungen an ein Folgesystem identifiziert werden. Zusätzliche Anforderungen wurden durch Anwenderbefragungen erhoben, weitere konnten im Zusammenhang mit kollaborativen Forschungsszenarien identifiziert werden. Mit Hilfe einer Videoanalyse konnten außerdem zusätzliche ‚implizite' Anforderungen identifiziert werden.

Es erfolgte die Konzeption des GHRC-Prototyps auf Basis der identifizierten Anforderungen. Diese Konzeption umfaßte unter anderem den Aufbau des Systems aus autonomen Laborstationen, die Konzeption der Stationensoftware und weiterer benötigter Applikationen. Für die Interaktion zwischen Systemkomponenten und Cryo-Devices wurden Dockingstationen und ihre modulare Anbindung mittels eines Device Integration Frameworks konzipiert. Ein Konzept zur lückenlosen Dokumentation der in einer Präparation vorhandenen Proben und ihrer Aliquoten wurde erarbeitet. Ein ergonomisches GUI wurde auf Basis einer Evaluierung ergonomischer Aspekte des Testsystems in Anlehnung an DIN EN ISO 9241 konzipiert.

Die Implementierung des GHRC-Prototyps auf Basis der erarbeiteten Konzeption wurde beschrieben. Die konzipierte Integration einer Workflow-Engine in die Stationensoftware autonomer Laborstationen konnte durch Verwendung der Windows Workflow Foundation realisiert werden. Durch die Trennung von Activity Code und XML-Workflow-Markup wurde die dezentrale Speicherung der Workflow-Definitionen erreicht. Die konzipierte modulare Anbindung von Laborgeräten an die Stationensoftware und ihre Steuerung aus dem Workflow heraus wurde durch ein eigenes Device Integration Framework und entsprechende Meta-Aktivitäten und geräteseitige Softwarekomponenten erreicht. Es leistet die Umsetzung abstrahierter Kommandos in gerätespezifische Syntax einer konkreten Geräteapplikation, so daß eine Austauschbarkeit von Workflow-Definitionen zwischen Laboratorien erreicht wird.

Datenstrukturen für die verschiedenen Cryo-Devices wurden konzipiert, um ihre Interaktion sowohl mit der elektronischen Lagertank-Infrastruktur wie auch mit dem GHRC-Prototypen zu ermöglichen.

9 Literaturverzeichnis

[Aisen 2002]
Aisen, E.G.; Medina, V.H.; Venturio, A. (2002): Cryopreservation and Post-Thawed Fertility of Ram Semen Frozen in Different Trehalose Concentrations. Theriogenology 57: 1801-1808.

[Anderson 1992]
Anderson, J.R. (1992): Automaticity and the ACT* theory. American Journal of Psychology 105: 165-180.

[Berry 1984]
Berry, D.C.; Broadbent, D.E. (1984): On the relationship between task performance and associated verbalizable knowledge. The quarterly journal of experimental psychology 36A: 209-231.

[Berry 1987]
Berry, D.C. (1987): The problem of implicit knowledge. Expert Systems 4, 3: 144-151.

[Berry 1987a]
Berry, D.C.; Broadbent, D.E. (1987): The combination of explicit and implicit learning processes in task control. Psychological research 49, 7: 7-15.

[Betensky 2000]
Betensky, R.A.; Connick, E.; Devers, J.; Landay, A.L.; Nokta, M.; Plaeger, S.; Rosenblatt, H.; Schmitz, J.L.; Valentine, F.; Wara, D.; Weinberg, A.; Lederman, H.M. (2000): Shipment impairs lymphocyte proliferative responses to microbial antigens. Clin Diagn Lab Immunol. 7, 5: 759-63.

[Bettendorf 2005]
Bettendorf, E.; Malenfant, C.; Chabannon, C. (2005): RFID Technology and Electronic Tags to Identify Cryopreserved Materials. Cell Preservation Technology 3, 2: 112-114.

[Berman 2007]
Berman, R.T (2007): Using C++ to Write Automation Controller Software. Journal of the Association for Laboratory Automation 12: 12-16.

[Bielanski 2000]
Bielanski, A.; Nadin-Davis, S.; Sapp, T.; Lutze-Wallace, C. (2000): Viral contamination of embryos cryopreserved in liquid nitrogen. Cryobiology 40: 110-116.

[Böhle 1995]
Böhle, F. (1995): Sozialwissenschaftliche Grundlagen des CeA-Ansatzes. In H. Martin (Hrsg.): CeA – Computergestützte erfahrungsgeleitete Arbeit: 17-30. Springer, Berlin.

[Bradley 2005]
Bradley, M.B.; Cairo, M.S. (2005): Cord Blood Immunology and Stem Cell Transplantation. Human Immunology 66, 5: 431-446.

[Büssing 1999]
Büssing, A.; Herbig, B.; Ewert, T. (1999): Implizites Wissen und erfahrungsgeleitetes Arbeitshandeln. Berichte aus dem Lehrstuhl für Psychologie der TU München 48. Kapitel 2: Konzepte.

[Büssing 2002]
Büssing, A.; Herbig, B.; Latzel, A. (2002): Das Zusammenspiel zwischen Erfahrung, implizitem und explizitem Wissen beim Handeln in kritischen Situationen. Berichte aus dem Lehrstuhl für Psychologie der TU München 66. Kapitel 2.3: Zusammenhänge zwischen den verschiedenen Formen des Wissens und Handelns.

[Cagindi 2004]
Cagindi, Ö; Ötles, S. (2004): Importance of laboratory information management systems (LIMS) software for food processing factories. Journal of Food Engineering 65: 565-568.

[Cahn 2007]
Cahn, M.H.; Russo, M.F. (2007): Python and Automated Laboratory System Control. Journal of the Association for Laboratory Automation (JALA) 12: 46-55.

[Carus 1996]
Carus, U. (1996): CeA – Computergestützte erfahrungsgeleitete Arbeit und Konsequenzen für die Berufsbildung. In R. Bremer (Hrsg.): Doppelqualifikation und Integration beruflicher und allgemeiner Bildung: 279-291.

[CDC 2007]
Centers for Disease Control and Prevention (2007): Progress in global measles control and mortality reduction 2000-2006. MMWR Morb Mortal Wkly Rep. 56, 47: 1237-41.

[Cedeno 2007]
Cedeno, W.; Laplante, P.A. (2007): An Overview of Real-Time Operating Systems. Journal of the Association for Laboratory Automation (JALA) 12: 40-45.

[Chain 1940]
Chain, E.; Florey, H.W.; Adelaide, M.B.; Gardner, A.D.; Heatley, H.G.; Jennings, M.A.; Orr-Ewing, J.; Sanders, A.G. (1940): Penicillin as a chemotherapeutic agent. Lancet 2: 226.

[Chen 1998]
Chen, V.W.; Holt, M.G.; Banerjee, R.L.; Thakur, D.S.; Leister, K.J. (1998): Systems Development Strategy: A Component-based approach: The Architecture. Proceedings of the International Symposium on Laboratory Automation and Robotics ISLAR, Boston, 1998.

[Dienes 1997]
Dienes, Z.; Berry, D. (1997): Implicit learning: below the subjective threshold. Psychonomic bulletin & review 4: 3-23.

[Disis 2006]
Disis, M.L.; dela Rosa, C.; Goodell, V.; Kuan, L.Y.; Chang, J.C.; Kuus-Reichel, K.; Clay, T.M.; Kim-Lyerly, H.; Bhatia, S.; Ghanekar, S.A.; Maino, V.C.; Maecker H.T. (2006): Maximizing the retention of antigen specific lymphocyte function after cryopreservation. Journal of Immunological Methods 308, 1-2: 13-18.

[Durst 2003]
Durst, C.H.P. (2003): Analyse der Anforderungen an Datenbanken für die Langzeitablage biologischer Zellen. Diplomarbeit, Universität des Saarlandes.

[Durst 2004]
Durst, C.H.P.; Ihmig, F.R.; Hotz, G.; Zimmermann, H. (2004): The demand for low temperature electronics in cell cryobank databases. Proc. 6th European Workshop on Low Temperature Electronics (WOLTE-6), ESTEC, Noordwijk, The Netherlands: 279-286.

[Durst 2008]
Durst, C. H. P.; Ihmig, F. R.; Shirley, S. G.; Fuchs, C. C.; Zimmermann, H. (2008): ChameleonLab®: A Workflow Management System for Biomedical Laboratories Based on Low Temperature Electronics. Proceedings of the 8th International Workshop on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008: 76-77.

[Elliott 2007]
Elliott, C.; Vijayakumar, V.; Zink, W.; Hansen, R. (2007): National Instruments LabView: A Programming Environment for Laboratory Automation and Measurement. Journal of the Association for Laboratory Automation (JALA) 12: 17-27.

[Felder 1998]
Felder, R.A. (1998): Modular workcells: modern methods for laboratory automation. Clinica Chimica Acta 278: 257-267.

[Fixemer 2003]
Fixemer, T.; Wissensbach, U.; Flockerzi, V.; Bonkhoff, H. (2003): Expression of the Ca^{2+}-selective cation channel TRPV6 in human prostate cancer: a novel prognostic marker for tumor progression. Oncogene 22, 49: 7858-61.

[Fleming 1929]
Fleming, A. (1929): On the antibacterial action of cultures of a penicillium, with special reference to their use in the isolation of B. influenzae. British Journal of Experimental Pathology 10: 226-236.

[Frischmuth 2002]
Frischmuth, N. (2002): Anreizsysteme für den innerbetrieblichen Wissensmarkt. Organisatorische und technologische Möglichkeiten. Tectum-Verlag, Marburg. Kapitel 3.2: Wissensarten.

[Fromme 2000]
Fromme, M.; Schröder, M. (2000): Elektronisches Laborbuch. Ergebnisbericht Forschung und Entwicklung 2000, Informationstechnik: 137-138. Hahn-Meitner-Institut Berlin GmbH (HMI).

[Fuchs 2008]
Fuchs, C.C.; Ihmig, F.R.; Shirley, S.G.; Zimmermann, H.: Development of a 32-bit microcontroller-based cryotank backplane system for use in the "Global HIV Vaccine Research Cryorepository". Proceedings of the 8th International Workshop on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008: 82-83.

[Gelman 1994]
Gelman, R. (1994): Constructivism and supporting environment. In D. Tirosh (Ed.): Implicit and explicit knowledge: an educational approach. Human development 6: 55-82.

[Gentsch 1993]
Gentsch, J. (1993): Flexible laboratory automation to meet the challenge of the ’90s: Chemometrics and Intelligent Laboratory Systems. Laboratory Information Management 21: 229-233.

[Gentsch 2000]
Gentsch, J.; Bruttger, O. (2000): High Throughput at the nano scale. Journal of the Association for Laboratory Automation (JALA) 5, 3: 60-65.

[Germann 2009]
Germann, A.; Durst, C.H.P.; Ihmig, F.R.; Shirley, S.G.; Schön, U.; Koch, S.; Schulz, J. C.; Schmidt, J.; Zimmermann, H.; Meyerhans, A.; Von Briesen, H. and the GHRC-Consortium (2009): Global HIV Vaccine Research Cryorepository-GHRC. Procedia in Vaccinology 1, 1: 49-62.

[GHAVE 2005]
The Global HIV/AIDS Vaccine Enterprise (2005): Scientific Strategic Plan. Coordinating Committee of the Global HIV/AIDS Vaccine Enterprise. PLoS Med. 2, 2: e25.

[Gibbon 1984]
Gibbon, G. (1984): Trends in laboratory information management systems. Trends in analytical chemistry 3, 2: 36-38.

[Gibbon 1996]
Gibbon, G. (1996): A brief history of LIMS. Laboratory Automation and Information Management 32, 1: 1-5.

[Gillespie 1997]
Gillespie, H. (1997): LIMS address new food regulations. Inside Lab Manag., October 1997.

[Graves 2000]
Graves, B.S.; Mifflin, T.A.; Kell, S.; Gunderson, J.; Geddy, S.; Felder, R.A. (2000): Software Implementation of Biological Repository for Human Genetic Material. Journal of the Association for Laboratory Automation (JALA) 5, 6: 106-108.

[Herbig 2001]
Herbig, B. (2001): Vergleichende Untersuchung von Struktur und Inhalt expliziten und impliziten Wissens im Arbeitskontext. Shaker-Verlag, Aachen. Kapitel 2.1: Fakten und Prozeduren bei implizitem und explizitem Wissen.

[Herbig 2001a]
Herbig, B. (2001): Vergleichende Untersuchung von Struktur und Inhalt expliziten und impliziten Wissens im Arbeitskontext. Shaker-Verlag, Aachen. Kapitel 2.8: Erfahrung, Expertise und Wissen.

[Herbig 2001b]
Herbig, B. (2001): Vergleichende Untersuchung von Struktur und Inhalt expliziten und impliziten Wissens im Arbeitskontext. Shaker-Verlag, Aachen. Kapitel 2.6: Fehlerhaftigkeit impliziten und expliziten Wissens.

[Hoffmann 1998]
Hoffmann, G.E. (1998): Concepts for the third generation of laboratory systems. Clinica Chimica Acta 278: 203-216.

[Hung 2004]
Hung, M.H.; Tsai, J.; Cheng, F.T.; Yang, H.C. (2004): Development of an Ethernet-based equipment integration framework for factory automation. Robotics and Computer-Integrated Manufacturing 20: 369-383.

[Ihmig 2003]
Ihmig, F.R.; Shirley, S.G.; Zimmermann, H. (2003): Evaluation and Adaption of Flash-Memory for Cryobiophysical Applications. Proc. 2nd VDE World Microtechnologies Congress, Munich, Germany: 643-648.

[Ihmig 2004]
Ihmig, F.R.; Shirley, S.G.; Zimmermann, H. (2004): Electronic Memory Devices for Cryobiological Applications. Proc. 6th European Workshop on Low Temperature Electronics (WOLTE-6), ESTEC, Noordwijk, The Netherlands: 153-160.

[Ihmig 2006]
Ihmig, F.R.; Shirley, S.G.; Durst, C.H.P.; Zimmermann, H. (2006): Cryogenic electronic memory infrastructure for physically related „continuity of care records" of frozen cells. Cryogenics 46, 4: 312-320.

[Ihmig 2006a]
Ihmig, F.R.; Shirley, S.G.; Durst, C.H.P.; Fuchs, C.C.; Blanck, M.; Zimmermann, H. (2006): Low-Temperature Electronics for the Cryopreservation of Living Cells.
Proc. of the 7th European Workshop on Low Temperature Electronics (WOLTE-7) in Noordwijk (Niederlande), 21.-23. 06 2006: 13-20.

[Ihmig 2008]
Ihmig, F. R.; Shirley, S. G; Fuchs, C. C.; Durst, C. H. P.; Schulz, J. C.; Zimmermann, H. (2008): Development of Electronic Cryovials and Electronic Cryotank Modules for the 'Global HIV Vaccine Research Cryorepository'. Proc. of the 8th International Workshop on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008: 80-81.

[Ihmig 2009]
Ihmig, F.R.; Shirley, S.G.; Durst, C.H.P.; Schulz, J.C.; Von Briesen, H.; Zimmermann, H. (2009): The technology of the Global HIV Vaccine Research Cryorepository. Engineering in Life Sciences, in press. Published Online: Sep 11 2009. DOI: 10.1002/elsc.200800121.

[Kirsner 1998]
Kirsner, K.; Speelman, C. (1998): Introduction and Overview. In Kirsner, K.; Speelman, C.; Maybery, M.; O`Brien-Malone, A.; Anderson, M.; MacLeod, C. (Eds.): Implicit and explicit mental processes. Erlbaum, Mawah, NJ: 135-147.

[Koch 1999]
Koch, S.; Mandl, H. (1999): Wissensmanagement - Anwendungsfelder und Instrumente für die Praxis. Forschungsbericht Nr. 103 des Lehrstuhls für Empirische Pädagogik und Pädagogische Psychologie der Ludwig-Maximilians-Universität, München. Kapitel 2.1: Bausteine des Wissensmanagements.

[Kruse 2004]
Kruse, C.; Birth, M.; Rohwedel, J.; Assmuth, K.; Goepel, A.; Wedel, T. (2004): Pluripotency of adult stem cells derived from human and rat pancreas. Applied Physics A. Published Online 26 May 2004, Springer-Verlag.

[Kuo 2002]
Kuo, C.H.; Juang, J.H.; Hsu, B.R.S.; Lu, W.T.; Yao, N.K. (2002): In Vitro and In Vivo Studies on Cryopreserved Pancreatic Islets. Transplantation Proceedings 34: 2693-2695.

[Langer 1999]
Langer, S.; Lau, D.; Eckardt, T.; Jahr, H.; Brandhorst, H.; Brandhorst, D.; Hering, B.J.; Federlin, K.; Bretzel, R.G. (1999): Viability and Recovery of Frozen-Thawed Human Islets and In Vivo Quality Control by Xenotransplantation. Journal of Molecular Medicine 77: 172-174.

[Laugwitz 2001]
Laugwitz, B. (2001): Experimentelle Untersuchung von Regeln der Ästhetik von Farbkombinationen und von Effekten auf den Benutzer bei ihrer Anwendung im Benutzungsoberflächendesign. Dissertation, Universität Mannheim.

[Leoni 2002]
Leoni, G.; Bogliolo, L.; Berlinguer, F.; Rosati, I.; Pintus, P.P.; Ledda, S.; Naitana, S. (2002): Defined Media for Vitrification, Warming and Rehydration: Effects on Post-Thaw Protein Synthesis and Viability of In Vitro Derived Ovine Embryos. Cryobiology 45: 204-212.

[Lewicki 1987]
Lewicki, P.; Czyzewska, M.; Hoffmann, H. (1987): Unconscious acquisition of complex procedural knowledge: Learning, Memory, and Cognition. Journal of experimental psychology 13: 523-530.

[Liberty 2002]
Liberty, J. (2002): Programmieren mit C#. Entwicklung von .NET-Applikationen. 1. Auflage. O'Reilly & Associates, Cambridge: 547-570.

[Lindner 1990]
Lindner, M. (1990): Laboratory automation and robotics – quo vadis? In Karjalainen, E.J. (Ed.): Scientific Computing and Automation (Europe) 1990. Elsevier, Amsterdam: 273-290.

[Mahaffey 1991]
Mahaffey, R.R. (1991): Information technology in the laboratory. Chemometrics and Intelligent Laboratory Systems: Laboratory Information Management 13: 69-74.

[Mandl 2000]
Mandl, H.; Reinmann-Rothmeier, G. (2000): Wissensmanagement. Die strategische Bedeutung des Wissensmanagements. Oldenbourg, München. Kapitel 1.8: Die zentrale Idee der Learning Communities.

[Massip 2001]
Massip, A. (2001): Cryopreservation of Embryos of Farm Animals. Reprod. Domest. Anim. 36, 2: 49-55.

[McIntosh 2003]
McIntosh, R.L.; Yau, A. (2003): A Flexible and Robust Peer-to-Peer Architecture with XML-Based Open Communication for Laboratory Automation. Journal of the Association for Laboratory Automation (JALA) 8, 1: 38-45.

[McIntosh 2004]
McIntosh, R.L. (2004): Open-Source Tools for Distributed Device Control within a Service-Oriented Architecture. Journal of the Association for Laboratory Automation (JALA) 9: 404-410.

[Meryman 1971]
Meryman, H. T. (1971): Absence of Unfrozen Freezable water in Rapidly Frozen Red Cells. Cryobiology 7, 4-6: 252-255.

[Mole 1993]
Mole, D.; Mason, R.J.; McDowall, R.D. (1993): The development of a strategy for the implementation of automation in a bioanalytical laboratory. Journal of Pharmaceutical & Biomedical Analysis 11, 3: 183-190.

[Muldrew 1999]
Muldrew, K.; McGann, L.E. (1999): Cryobiology - A Short Course. http://www.ucalgary.ca/~kmuldrew/cryo_course/course_outline.html. Letzter Zugriff am 23.03.2009.

[Müller 2004]
Müller, M.; Herbig, B. (2004): Methoden zur Erhebung und Abbildung impliziten Wissens. Ergebnisse einer Literaturrecherche. Berichte aus dem Lehrstuhl für Psychologie der TU München 74. Kapitel 3: Theoretische Überlegungen zur Erhebung und Vermittlung impliziten Wissens.

[Muller 1999]
Muller, E.; Bassin, M.; Troyon, J.P.; Novak, P. (1999): Implementation of rapid result management systems in the metals industry. Laboratory Automation and information management 34: 31-39.

[Mulzer 2006]
Mulzer, K. (2006): Sprachverständnis und implizites Wissen. In Knoepffler, N.; Vossenkuhl, W.; Peetz, S.; Lauth, B. (Hrsg.): Münchener philosophische Beiträge, Band 18. Herbert Utz, München. Kapitel 4: Wissen, Können und Verstehen.

[Myers 1992]
Myers, C.; Davids, K. (1992): Knowing and doing. Tacit skills at work. Personnel management 24, 2: 45-47.

[Neuweg 2004]
Neuweg, G.H. (2004): Könnerschaft und implizites Wissen. Zur lehr- lerntheoretischen Bedeutung der Erkenntnis- und Wissenstheorie Michel Polanyis. 3. Auflage. Waxmann, Münster. Kapitel 8.3: Die Theorie des impliziten Wissens im Aufriß.

[Neuweg 2004a]
Neuweg, G.H. (2004): Könnerschaft und implizites Wissen. Zur lehr- lerntheoretischen Bedeutung der Erkenntnis- und Wissenstheorie Michel Polanyis. 3. Auflage. Waxmann, Münster. Annäherungen an das Forschungsfeld.

[Neuweg 1998]
Neuweg, G.H. (1998): Self-reference and the loss of meaning. Appraisal 2, 1: 37-42.

[Nonaka 1997]
Nonaka, I.; Takeuchi, H. (1997): Die Organisation des Wissens. Wie japanische Unternehmen eine brachliegende Ressource nutzbar machen. Campus, Frankfurt. Kapitel 3: Theorie der Wissensschaffung im Unternehmen.

[Nonaka 2000]
Nonaka, I.; Toyama, R.; Konno, N. (2000): SECI, Ba and Leadership: a Unified Model of Dynamic Knowledge Creation. Long Range Planning 33, 1: 5-34.

[Oberauer 1993]
Oberauer, K. (1993): Prozedurales und deklaratives Wissen und das Paradigma der Informationsverarbeitung. Sprache & Kognition 12: 30-43.

[Ölund 2007]
Ölund, G.; Lindqvist, P.; Litton, J.E. (2007): BIMS: An information management system for biobanking in the 21st century. IBM Systems Journal 46, 1: 171-182.

[PCT 2008]
Perma Cryo Technologie GmbH: Innovationen bei -196°C. http://www.perma-cryo.com/fileadmin/user_upload/PCT_Folder_SouthAfrica_DE_web.pdf. Letzter Zugriff 14.09.2009.

[Pearson 2004]
Pearson, H. (2004): Summit calls for clear view of deposits in all biobanks. Nature 432: 426.

[Polanyi 1966]
Polanyi, M. (1966): The Tacit Dimension. Doubleday, Garden City, New York. Kapitel 1: Tacit Knowing.

[Polanyi 1985]
Polanyi, M. (1985): Implizites Wissen. Suhrkamp, Frankfurt am Main. Kapitel 1: Implizites Wissen.

[Polge 1949]
Polge, C.; Smith, A.U.; Parkes, A.S. (1949): Revival of spermatozoa after vitrification and dehydration at low temperatures. Nature 164: 666.

[Probst 1998]
Probst, G.; Raub, S.; Romhardt, K. (1998): Wissen managen. Wie Unternehmen ihre wertvollste Ressource optimal nutzen. Gabler, Wiesbaden. Kapitel 3: Bausteine des Wissensmanagements.

[Rajotte 1999]
Rajotte, R. V. (1999): Islet Cryopreservation Protocols. Annals of the New York Academy Of Sciences 875: 201-207.

[Reber 1989]
Reber, A.S. (1989): Implicit learning and tacit knowledge. Journal of experimental psychology 118: 219-235.

[Reber 1977]
Reber, A.S.; Lewis, S. (1977): Implicit learning. An analysis of the form and structure of a body of tacit knowledge. Cognition 5: 333-361.

[Reinmann-Rothmeier 1998]
Reinmann-Rothmeier, G; Mandl, H. (1998): Wissensvermittlung: Ansätze zur Förderung des Wissenserwerbs. In Klix, F.; Spada, H. (Hrsg.): Wissen (Enzyklopädie der Psychologie: Themenbereich C, Theorie und Forschung: Ser. 2 Kognition) 6: 457-490. Hogrefe, Göttingen.

[Reinmann-Rothmeier 2000]
Reinmann-Rothmeier, G; Mandl, H. (2000): Individuelles Wissensmanagement – Strategien für den persönlichen Umgang mit Information und Wissen am Arbeitsplatz. Huber, Göttingen. Kapitel 1.2.: Ein Referenzmodell zum Wissensmanagement.

[Rodziewicz 2004]
Rodziewicz, P.; Bell, B. (2004): Overview and Architecture of the Java Integration Framework, Hybrid Scheduler, and Web-Enabled LIMS. Journal of the Association for Laboratory Automation (JALA) 9: 411-420.

[Ryle 1949]
Ryle, G. (1949): The Concept of mind. Hutchinson, London. Kapitel 2: Knowing how and knowing that.

[Sarkozi 2003]
Sarkozi, L.; Simson, E.; Ramanathan, L. (2003): The effects of total laboratory automation on the management of a clinical chemistry laboratory. Retrospective analysis of 36 years. Clinica Chimica Acta 329: 89-94.

[Sasaki 1998]
Sasaki, M.; Kageoka, T.; Ogura, K.; Kataoka, H.; Ueta, T.; Sugihara, S. (1998): Total laboratory automation in Japan – Past, Present and the future. Clinica Chimica Acta 278: 217-227.

[Schäfer 2004]
Schäfer, R. (2004): Concepts for Dynamic Scheduling in the Laboratory. Journal of the Association for Laboratory Automation (JALA) 9: 382-397.

[Scheer 2002]
Scheer, A.W. (2002): ARIS – Vom Geschäftsprozeß zum Anwendungssystem. 4., durchgesehene Auflage. Springer, Berlin. Kapitel D.I.3: Wissensmanagement.

[Scheer 2002a]
Scheer, A.W. (2002): ARIS – Vom Geschäftsprozeß zum Anwendungssystem. 4., durchgesehene Auflage. Springer, Berlin. Kapitel D.III: Workflowsteuerung.

[Schröder 2008]
Schröder, M.; Denzer, V.(2008): ELAB 2.0 – Flexibilität und Archivierung. Neuigkeiten und Wissenswertes aus der Zentralen Datenverarbeitung. Info FMD, Oktober 2008.

[Seger 1994]
Seger, C.A. (1994): Implicit Learning. Psychological Bulletin 115: 163-196.

[Sekaly 2008]
Sekaly, R.P. (2008): The failed HIV Merck vaccine study: a step back or a launching point for future vaccine development? J Exp Med. 205, 1: 7-12.

[Shirley 2006]
Shirley, S.G.; Ihmig, F.R.; Zimmermann, H.: Prototype Electronic Infrastructure for a Cryo-repository. Proc. of the 7th European Workshop on Low Temperature Electronics (WOLTE-7) in Noordwijk (Niederlande), 21.-23.06 2006: 263-268.

[Shirley 2009]
Shirley, S.G.; Durst, C.H.P.; Fuchs, C.C.; Zimmermann, H.; Ihmig, F.R. (2009): A large-scale cryoelectronic system for biological sample banking. Cryogenics, in press. Published online 19 January 2009. DOI: 10.1016/j.cryogenics.2008.12.022.

[Sommerfeld 1999]
Sommerfeld, V.; Niemann, H. (1999): Cryopreservation of Bovine in Vitro Produced Embryos using Ethylene Glycol in Controlled Freezing or Vitrification. Cryobiology 38: 95-105.

[Speelman 1998]
Speelman, C. (1998): Implicit expertise: do we expect too much from our experts? In Kirsner, K.; Speelman, C.; Maybery, M.; O`Brien-Malone, A.; Anderson, M.; MacLeod, C. (Eds.): Implicit and explicit mental processes. Erlbaum, Mawah, NJ: 135-147.

[Staab 1999]
Staab, T.A.; Elling, J.W. (1999): ASTM E1989-98 - The New Instrument Control Standard. Journal of the Association for Laboratory Automation (JALA) 4, 3: 40-42.

[Tanenbaum 1995]
Tanenbaum, A.S. (1995): Moderne Betriebssysteme. 2. Auflage, Hanser. Kapitel 2: Prozesse und Threads.

[Tatsumi 1999]
Tatsumi, N.; Okuda, K.; Tsuda, I. (1999): A new direction in automated laboratory testing in Japan: five years of experience with total laboratory automation system management. Clinica Chimica Acta 290: 93-108.

[Thakur 1998]
Thakur, D.S.; Chen, V.W.; Banerjee, R.L.; Holt, M.G.; Leister, K.J. (1998): Systems Development Strategy: A Component-based approach: The Overview. Proceedings of the International Symposium on Laboratory Automation and Robotics ISLAR, Boston, 1998.

[Thobe 2003]
Thobe, W. (2003): Externalisierung impliziten Wissens. Ein verhaltenstheoretisch fundierter Beitrag zum organisationalen Lernen. Peter Lang, Frankfurt am Main.

[Thomas 1996]
Thomas, M. J. G.; Parry, E.S.; Nash, S.G.; Bell, S.H. (1996): A Method for the Cryopreservation of Red Blood Cells using Hydroxyethyl Starch as Cryoprotectant. Transfusion Science 17: 385-396.

[Tijssen 2008]
Tijssen, M.R.; Woelders, H.; de Vries-van Rossen, A.; van der Schoot, C.E.; Voermans, C.; Lagerberg, J.W. (2008): Improved postthaw viability and in vitro functionality of peripheral blood hematopoietic progenitor cells after cryopreservation with a theoretically optimized freezing curve. Transfusion, 2008 Feb 21 [Epub ahead of print].

[Weinberg 2000]
Weinberg, A.; Wohl, D.A.; Brown, D.G.; Pott, G.B.; Zhang, L.; Ray, M.G.; van der Horst, C. (2000): Effect of cryopreservation on measurement of cytomegalovirus-specific cellular immune responses in HIV-infected patients. J Acquir Immune Defic Syndr. 25, 2: 109-14.

[Whelan 2004]
Whelan, K.E.; King, R.D. (2004): Intelligent software for laboratory automation. Trends in Biotechnology 22, 9.

[Witt 1996]
Witt, H. (1996): Erfahrung und Intuition vs. Wissen und Kognition: Überlegungen zu einem besonderen Modus der Handlungssteuerung. In Witruk, E.; Friedrich, G. (Hrsg.): Pädagogische Psychologie im Streit um ein neues Selbstverständnis. Bericht über die 5. Tagung der Fachgruppe für pädagogische Psychologie der Deutschen Gesellschaft für Psychologie. Verein empirische Pädagogik, Leipzig.

[Wood 1993]
Wood, M.D.; Franchetti, J.A. (1993): Laboratory automation using robotics and information management systems. Current opinion in Biotechnology 4: 91-94.

[Wyatt 1996]
Wyatt, A.J. (1996): An automated sample preparation and testing laboratory for North West Water. Laboratory Automation and Information Management 32: 103-108.

[Wusteman 2002]
Wusteman, M. C.; Pegg, D.E.; Robinson, M.P.; Wang, L.H.; Fitch, P. (2002). Vitrification Media: Toxicity, Permeability and Dielectric Properties. Cryobiology 44: 24-37.

[Yavilevich 2002]
Yavilevich, M. (2002): A New approach in Laboratory Automation: Fast spin, a new pre-analytical station. Journal of the Association for Laboratory Automation (JALA) 7, 1: 89-93.

[Zimmermann 2004]
Zimmermann, H.; Ihmig, F.R.; Shirley, S.G. (2004): A low cost stage for testing electronics between room and liquid nitrogen temperatures. Proc. 6th European Workshop on Low Temperature Electronics (WOLTE-6), ESTEC, Noordwijk, The Netherlands: 287-293.

10 Anhang

10.1 Abkürzungsverzeichnis

Abkürzung	*Bedeutung*	*Kapitel / Abschnitt*
AIDS	Acquired ImmunoDeficiency Syndrome	4.2
BPEL	Business Process Execution Language	2.2.4
BPML	Business Process Modeling Language	2.2.4
BPMN	Business Process Modelling Notation	2.2.4
CAVD	Collaboration for AIDS Vaccine Discovery	4.2
CAQ-System	Computer Aided Quality Assurance System	3.1
CIL	Common Intermediate Language	5.5.1.3
DCMS	Device Control Management System	5.5.2
DMSO	Dimethylsulfoxid	2.2.1
ECM	Enterprise-Content-Management-System	3.1
EDMS	Elektronisches Dokumenten-Management-System	3.1
EMID	Elektromagnetische ID	3.2
GALP	Good Automated Laboratory Practice	3.3.3
GCLP	Good Clinical Laboratory Practice	1
GHAVE	Global HIV/AIDS Vaccine Enterprise	4.2
GHRC	Global HIV Specimen Research Cryorepository	4.2
GLP	Good Laboratory Practice	3.3.1
GMP	Good Manufacturing Practice	1
GUI	Graphical User Interface	5.4.6
HIV	Humanes Immundefizienz-Virus	4.2
JEDEC	Joint Electron Device Engineering Council	5.5.2.2
LECIS	Laboratory Equipment Control Interface Standard	3.4.2.1
LIMS	Laborinformations-Management-System	3.3.3
PBMCs	Mononukleäre Immunzellen	1
PCB	Printed Circuit Board	4.3
PLS	Prozessleitsystem	3.3.1
QM	Qualitätsmanagement	3.1
QA	Quality Assurance	4.1
RFID	Radio Frequency Identification	3.2
RTOS	Real Time Operating Systems, Echtzeitbetriebssysteme	3.4.1.2
SOPs	Standard Operating Procedures	3.1
SPI	Serial Peripheral Interface	4.3
TLA	Total Lab Automation	3.3.2.1
VM	Virtuelle Maschine	5.5.1

Abkürzung	*Bedeutung*	*Kapitel / Abschnitt*
WfMS	Workflow-Management-System	2.2.4
WMS	Workflow-Management-System	2.2.4
WWF	Windows Workflow Foundation	5.5.1.2
XML	Extensible Markup Language	5.2.1
XOML	Extensible Object Markup Language	5.5.1.3
XPDL	XML Process Definition Language	2.2.4

10.2 Verweisverzeichnis

10.3 Veröffentlichungen

Durst, C. H. P.; Ihmig, F. R; Hotz, G.; Zimmermann, H. (2004): The Demand for Low Temperature Electronics in Cell Cryobank Databases.
Proceedings of the Sixth European Workshop on Low Temperature Electronics (WOLTE-6). ESTEC in Noordwijk (Niederlande), 23.–25.06.2004: 279-286.

Zimmermann, H.; Katsen, A. D.; Ihmig, F. R.; Durst, C. H. P.; Shirley, S. G.; Fuhr, G. R. (2004): First Steps of an Interdisciplinary Approach towards Miniaturised Cryopreservation for Cellular Nanobiotechnology.
IEEE Proceedings–Nanobiotechnol. 151, 4: 134-138.

Durst, C.H.P.; Ihmig, F.R.; Biel, M.; Daffertshofer, M.; Zimmermann, H. (2005): XML based Process Management in Cryo-Biotechnology: The ChameleonLab.
CEUR Workshop Proceedings 145: 55-64.

Durst, C.H.P.; Ihmig, F.R.; Biel, M.; Daffertshofer, M.; Zimmermann, H. (2005): XML based Process Management in Cryo-Biotechnology: The ChameleonLab.
Vortrag anlässlich der 11th Conference Business, Technology and Web (BTW2005)
in Karlsruhe (Baden-Württemberg), 01.03.2005.

Ihmig, F.R.; Shirley, S.G.; Durst, C.H.P.; Zimmermann, H. (2006): Cryogenic Electronic Memory Infrastructure for Physically Related Continuity of Care Records of Frozen Cells.
Cryogenics 46, 4: 312-320.

Durst, C.H.P.; Ihmig, F.R.; Biel, M.; Daffertshofer, M.; Zimmermann, H. (2006): A Method and Infrastructure for Long-Term Managing of Sample Preparation Knowledge for Cryobiomedical Applications.
Proceedings of the 6th International Conference on Knowledge Management (I-KNOW ’06) in Graz (Österreich), 06.-08.09.2006: 359-366.

Ihmig, F.R.; Shirley, S.G.; Durst, C.H.P.; Fuchs, C.C.; Blanck, M.; Zimmermann, H. (2006): Low-Temperature Electronics for the Cryopreservation of Living Cells.
Proceedings of the 7th European Workshop on Low Temperature Electronics (WOLTE-7) in Noordwijk (Niederlande), 21.-23. 06 2006: 13-20.

Zimmermann, H.; Ihmig, F.R.; Katsen-Globa, A.; Ehrhart, F.; Durst, C.H.P.; Shirley, S.G. (2006): Cryobiotechnology – Low Temperature Microsystems for Biotechnology and Regenerative Medicine. Microsystems Technology 2006: 22-25.

Durst, C.H.P.; Ihmig, F.R.; Ehrhart, F.; Biel, M.; Daffertshofer, M.; Zimmermann, H. (2006): A Method and Technology for Reliable Sample-Controlled Execution of Preparation and Freezing Protocols in Biomedical Laboratories and Cryobanks.
Abstractbook des 43rd Meeting of the Society for Cryobiology in Association with the Society for Low Temperature Biology in Hamburg, 24.-27.07.2006: 152.

Durst, C.H.P.; Ihmig, F.R.; Ehrhart, F.; Biel, M.; Daffertshofer, M.; Zimmermann, H. (2006): A Method and Technology for Reliable Sample-Controlled Execution of Preparation and Freezing Protocols in Biomedical Laboratories and Cryobanks.
Posterbeitrag anlässlich des 43rd Meeting of the Society for Cryobiology in Association with the Society for Low Temperature Biology in Hamburg, 24.-27.07.2006.

Durst, C.H.P.; Ihmig, F.R.; Shirley S.G.; Zimmermann, H. (2007): Management, Interchange and Reproducible Execution of Sample Preparation Knowledge in Collaborative Research Scenarios.
Proceedings of the 7th International Conference on Knowledge Management (I-KNOW ‘07) in Graz (Österreich), 05.-07.09.2007: 111-117.

Durst, C. H. P.; Ihmig, F. R.; Shirley, S. G.; Zimmermann, H. (2007): ChameleonLab: A Preparation Knowledge and Workflow Management System for Cryobiomedical Laboratories and Biobanks.
Proceedings der 41. DGBMT-Jahrestagung in Aachen (Nordrhein-Westfalen), 26.-29.09.2007.

Durst, C. H. P.; Ihmig, F. R.; Shirley, S. G.; Zimmermann, H. (2007): ChameleonLab: A Preparation Knowledge and Workflow Management System for Cryobiomedical Laboratories and Biobanks.
Posterbeitrag anlässlich der 41. DGBMT-Jahrestagung in Aachen (Nordrhein-Westfalen), 26.-29.09.2007.

Durst, C. H. P.; Ihmig, F. R.; Shirley, S. G.; Fuchs, C. C.; Zimmermann, H. (2008): ChameleonLab®: A Workflow Management System for Biomedical Laboratories Based on Low Temperature Electronics.
Proceedings of the 8th International Workshop on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008: 76-77.

Durst, C. H. P.; Ihmig, F. R.; Shirley, S. G.; Fuchs, C. C.; Zimmermann, H. (2008): ChameleonLab®: A Workflow Management System for Biomedical Laboratories Based on Low Temperature Electronics.

Posterbeitrag anlässlich des 8th International Workshops on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008.

Ihmig, F. R.; Shirley, S. G; Fuchs, C. C.; Durst, C. H. P.; Schulz, J. C.; Zimmermann, H. (2008): Development of Electronic Cryovials and Electronic Cryotank Modules for the 'Global HIV Vaccine Research Cryorepository'.
Proceedings of the 8th International Workshop on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008: 80-81.

Ihmig, F. R.; Shirley, S. G; Fuchs, C. C.; Durst, C. H. P.; Schulz, J. C.; Zimmermann, H. (2008): Development of Electronic Cryovials and Electronic Cryotank Modules for the 'Global HIV Vaccine Research Cryorepository'.
Posterbeitrag anlässlich des 8th International Workshops on Low Temperature Electronics (WOLTE-8) in Jena/Gabelbach (Thüringen), 22.-25.06.2008.

Schulz, J. C.; Ihmig, F. R.; Durst, C. H. P.; Shirley, S. G.; Reich, A.; Germann, A.; Noah, C.; Korioth, F.; Kromminga, A.; Climaco, M.; Zimmermann, H.; Von Briesen, H. (2008): New Modular HIV-Cryosubstrates for Secure and Reliable Sample Identification and DMSO Reduction in Cryopreserved Peripheral Blood Mononuclear Cells.
AIDS Research and Human Retroviruses. October 2008, 24(s1): 63.

Shirley, S.G.; Durst, C.H.P.; Fuchs, C.C.; Zimmermann, H.; Ihmig, F.R. (2009): A large-scale cryoelectronic system for biological sample banking.
Cryogenics, in press. Published online 19 January 2009.
DOI: 10.1016/j.cryogenics.2008.12.022.

Ihmig, F.R.; Shirley, S.G.; Durst, C.H.P.; Schulz, J.C.; Von Briesen, H.; Zimmermann, H. (2009): The technology of the Global HIV Vaccine Research Cryorepository.
Engineering in Life Sciences, in press. Published Online: Sep 11 2009.
DOI: 10.1002/elsc.200800121.

Germann, A.; Durst, C.H.P.; Ihmig, F.R.; Shirley, S.G.; Schön, U.; Koch, S.; Schulz, J. C.; Schmidt, J.; Zimmermann, H.; Meyerhans, A.; Von Briesen, H. and the GHRC-Consortium (2009): Global HIV Vaccine Research Cryorepository-GHRC.
Procedia in Vaccinology 1, 1: 49-62.

10.4 Danksagung

Ich danke Herrn Prof. Dr. Günter Fuhr, dem Direktor des Fraunhofer-IBMT, für die Möglichkeit zur Promotion in einem sehr innovativen Umfeld, für kreative Diskussionen, für wichtige Einschätzungen und für das entgegengebrachte Vertrauen, meine Ideen innerhalb eines internationalen Projektes umsetzen zu dürfen.

Mein aufrichtiger Dank gilt Herrn Prof. Dr. Dr. h.c. mult. Günter Hotz für die Betreuung der vorliegenden Arbeit von Seiten der Universität des Saarlandes. Ich danke ihm für zahlreiche wertvolle Anregungen und für konstruktive Diskussionen auf Basis seines umfassenden Wissens und seiner großen Erfahrung. Es ist mir eine Ehre, einer seiner letzten Doktoranden sein zu dürfen.

In besonderem Maß danke ich Herrn Prof. Dr. Heiko Zimmermann, dem Leiter der Hauptabteilung ‚Biophysik und Kryotechnologie', für die Betreuung meiner Arbeit von Seiten des Fraunhofer-IBMT. Sein Interesse an der Thematik hat durch viele kreative, teils kontroverse Diskussionen letztendlich konstruktiv zum Gelingen des Systems und der vorliegenden Arbeit beigetragen.

Bei Herrn Prof. Dr. Manfred Schmitt vom Lehrstuhl für Molekular- und Zellbiologie der Universität des Saarlandes bedanke ich mich für die wissenschaftliche Begleitung der Arbeit.

Für die Erforschung und Entwicklung der Tieftemperaturelektronik, die den technologischen Rahmen dieser Arbeit darstellt und ohne die eine Umsetzung meiner Ideen nicht möglich gewesen wäre, danke ich Herrn Dr. Frank Ihmig, dem Leiter der Arbeitsgruppe ‚Kryomechatronik' am Fraunhofer-IBMT, Herrn Dr. Stephen Shirley und Dipl.-Kfm. Dipl.-Ing. Christian Fuchs. Ich danke ihnen weiterhin für die konstruktive, interdisziplinäre Zusammenarbeit.

Im Zusammenhang mit der Implementierung des ersten Testsystems bedanke ich mich bei Evotec Technologies GmbH (mittlerweile PerkinElmer zugehörig). Für die Zusammenarbeit bei der Implementierung des GHRC-Prototyps danke ich Dipl.-Ing. Kai Diercks, Oliver Borchers und Heiko Grade von Soventec GmbH. Für die Konstruktion der Substrate und Dockingstationen bedanke ich mich bei Marco Climaco und Perma Cryo Technologie GmbH.

Zum Gelingen der Arbeit haben in bedeutendem Maße auch Biologen und Technische Assistentinnen beigetragen. Sie haben spezifische Anforderungen definiert und iterativ an der Systemoptimierung mitgewirkt. In diesem Zusammenhang danke ich besonders PD Dr. Hagen von Briesen, Dr. Thomas Fixemer, Dr. Anja Germann, Dipl.-Biol. Julia Schulz, Susan

Zöllner, Anja Reich, Beatrice Kemp-Kamke, Dipl.-Biol. Mark Wossidlo und Dr. Alexandra Schütz.

Für die Förderung meiner Arbeit im Rahmen des GHRC-Projektes danke ich der Bill & Melinda Gates-Foundation.