
Meta Resource Management System

Requirements Specification

René Freude (707168) <rene.freude@gmx.de>
Daniel Sadilek (707297) <sadilek@tfh-berlin.de>
Stephan Weiß (706830) <steph.weiss@web.de>

Copyright © 2003

2003-04-21

Versionsgeschichte

Version 0.1

2003-04-07

Initial public release as project report.

Version 0.2

2003-04-21

Transformed and extended to requirements specification.

Dieses Pflichtenheft beschreibt die Motivation und Zielsetzung des Projektes "Meta Resource Management System" (MRMS). Es werden eine erste Analyse der Anforderungen unter Berücksichtigung des potentiellen Einsatzgebiets, der Arbeitsumgebung sowie der Datenumfänge vorgestellt und die einzusetzenden Techniken und Entwicklungswerkzeuge spezifiziert. Auf den daraus gewonnenen Erkenntnissen basierend wird eine Abschätzung über Aufwand und Machbarkeit des Vorhabens realisiert.

Das Softwareprojekt wird im Rahmen des Kurses "Softwareprojekte" (SWP) an der Technischen Fachhochschule Berlin von der Projektgruppe umgesetzt.

Inhaltsverzeichnis

1. Motivation	2
2. Zielbestimmung	2
2.1. Muss-Kriterien	2
2.2. Kann-Kriterien	3
2.3. Abgrenzungskriterien	4
3. Funktionalität	4
3.1. Muss-Funktionen	4
3.2. Kann-Funktionen	5
4. Nichtfunktionale Anforderungen	5
4.1. Leistungen	5
4.2. Benutzungsoberfläche	6
4.3. Qualitätsziele	6
4.4. Abgrenzung	6
5. Einsatz	7
5.1. Anwendungsbereiche	7
5.2. Zielgruppen	7
5.3. Betriebsbedingungen	7
6. Umgebung	7
6.1. Software	7
6.2. Hardware	7
6.3. Orgware	8
7. Daten und Umfänge	8
8. Einzusetzende Technologien	8
8.1. Zur Laufzeit einzusetzende Technologien	8
8.2. Entwicklungswerkzeuge und -techniken	9
9. Machbarkeit	10

1. Motivation

In jedem Unternehmen müssen die Ressourcen der alltäglichen Prozesse - wie Räume, Rechner, Mitarbeiter, Telefone, Software-Lizenzen, usw. - verwaltet werden. In einem Unternehmen mit wenigen Mitarbeitern erledigt dies häufig der Geschäftsführer selbst oder die Sekretärin. Wächst das Unternehmen mit der Zeit, so wird auch der Verwaltungsaufwand größer, so dass er nicht mehr "nebenher" von genannten Personen alleine zu bewerkstelligen ist. Einzelne Aufgabenbereiche werden dann an bestimmte Mitarbeiter delegiert; so könnte es zum Beispiel einen Mitarbeiter geben, der sich um die Beschaffung und Verwaltung der Software-Lizenzen kümmert, ein anderer wird vielleicht für neue Mitarbeiter die Rechner zusammenstellen und deren Bestand verwalten.

Diese historisch gewachsenen Strukturen werden immer ineffizienter, je größer das Unternehmen wird: Die Ressourcen werden dezentral von verschiedenen Mitarbeitern verwaltet; der eine hat z.B. eine handschriftliche Liste, der andere führt eine Excel-Tabelle. Datenkonsistenz zwischen den einzelnen Ressourcenlisten ist also nicht gewährleistet und Prozesse sind nicht oder nur informell definiert.

Folgender beispielhafter Prozess möge die Probleme veranschaulichen: „Ein neuer Mitarbeiter kommt ins Unternehmen“. Dieser muss in die Personalliste eingetragen werden - das macht die Sekretärin. Er benötigt einen Raum mit einem Arbeitsplatz - darum kümmert sich der Raumbeauftragte, nachdem er die Personalnummer erfahren hat, damit er den Mitarbeiter seinem neuen Raum zuordnen kann. Weiterhin braucht er einen Rechner mit entsprechender Software - dafür muss der Rechnerbeauftragte und anschließend der Verwalter der Software-Lizenzen kontaktiert werden. Vielleicht braucht er auch ein eigenes Telefon, ein Dienst-Fahrrad, eine Chipkarte für die Zutrittskontrolle und eine andere für die Kantine, usw.. Weitere Beispiele für Prozesse, an denen mehrere Ressourcenbeauftragte beteiligt sind: „Mitarbeiter zieht in einen anderen Raum um“, „Mitarbeiter benötigt einen schnelleren Rechner“ oder „Mitarbeiter benötigt neue Software“.

Sind die für solche Prozesse notwendigen Schritte und Kommunikationswege nicht definiert, so können leicht Fehler passieren; neue Mitarbeiter wissen nicht, an wen sie sich zu wenden haben und ist derjenige, der weiß, was für einen bestimmten Prozess alles zu tun ist, gerade erkrankt, ist der gesamte Arbeitsablauf blockiert oder zumindest verzögert.

2. Zielbestimmung

Unsere Vision ist ein Ressourcen Management System, dass ein ganzheitliches und effizientes Verwalten aller Ressourcen in einem Unternehmen ermöglicht, die an geschilderte o.ä. Prozesse geknüpft sind.

2.1. Muss-Kriterien

- *Verwaltung der Ressourcen in Ressourcenlisten*

Die Verwaltung von Ressourcen stellt die Kernanforderung an das zu entwickelnde System dar. Dieser Sachverhalt findet sich auch in der Namensgebung "Meta Resource Management System" wieder. So muss eine Abbildung von realen Konstellationen insofern stattfinden können, als dass konkrete Ressourcen aneinander gebunden werden können. Beispielsweise wird ein Tisch innerhalb des Systems an den Raum gebunden, in dem er sich in der Realität befindet.

- *Datenkonsistenz zwischen den Ressourcenlisten wird sichergestellt*

Grundlage der Bindung von Ressourcen ist eine Menge von Regeln, die den Gesetzmäßigkeiten der realen Welt bzw. realer Organisationsparadigmen entsprechen. Die Einhaltung dieser Regeln muss vom System gewährleistet werden. Entsprechend dürfen im Zuge von Interaktions- und Betriebsprozessen Informationen hinsichtlich der zu verwaltenden Ressourcen und ihrer Abhängigkeiten nicht verloren gehen bzw. ungewollt manipuliert werden.

- *Definition der Ressourcentypen*

Da sich die zu verwaltenden Ressourcentypen von Unternehmen zu Unternehmen unterscheiden, soll weiterhin jedem Unternehmen die Möglichkeit geboten werden, diese nach Bedarf selbst zu definieren. Das bedeutet, dass das System auch solche Use Cases unterstützen muss, die sich auf die Erstellung der Metadaten beziehen. Aus diesem Grund trägt das System das Wort "Meta" im Namen.

- *Benutzer- und Rechteverwaltung*

In einem Unternehmen haben die Mitarbeiter entsprechend ihrer Verantwortlichkeitsbereiche unterschiedliche Befugnisse beim Einsehen und Bearbeiten von Geschäftsdaten. Dieser Aspekt soll mit einem Benutzer- und Rechte-Management unterstützt und der Schutz sensibler Daten gewährleistet werden.

2.2. Kann-Kriterien

- *Modifikation der Ressourcentypen*

Das System kann die Veränderung von Ressourcentypen unterstützen und auf diese Weise Anpassbarkeit garantieren für den Fall, dass sich die Art der zu verwaltenden Ressourcen verändert. Im Gegensatz zum Muss-Kriterium „Definition von Ressourcentypen“ schließt dieses Kriterium auch solche Use Cases ein, die sich auf die Bearbeitung der Metadaten beziehen.

- *Unterstützung von Prozessen*

Unter einem Prozess ist in diesem Kontext die definierte Abfolge von einzelnen Arbeitsschritten zum Erreichen eines bestimmten Arbeitszieles zu verstehen. Dabei können unterschiedliche Personen mit unterschiedlichen Verantwortlichkeitsbereichen beteiligt sein. Die Unterstützung einer solchen prozessbasierten Organisationsstruktur ist wünschenswert. Es lässt sich zwischen folgenden Teilkriterien unterscheiden:

- *Definition von Prozessen*

Einem Anwender muss die Möglichkeit gegeben werden, die Arbeitsschritte, die einen Prozess in seiner Gesamtheit spezifizieren, in dem System abbilden zu können.

- *Automatisierung von Prozessschritten*

Es gibt Arbeitsschritte in Prozessen, die automatisierbar sind. Aufgrund von bestimmten Ereignissen können solche Arbeitsschritte ohne weitere Aktionen seitens der Benutzer durch das System vollzogen werden. Ein Beispiel hierfür ist die automatische Zuteilung einer Zugangskarte an einen neu angelegten Mitarbeiter.

- *Kommunikationsfunktion*

Da ein Prozess während seiner Lebensdauer zum Teil von unterschiedlichen Personen bearbeitet wird, kann zum Beispiel eine Möglichkeit bereitgestellt werden, Mitarbeiter dann zu informieren, wenn diese einen nächsten Arbeitsschritt ausführen sollen.

- *Reservierung von Ressourcen*

Um die Verwendung von Ressourcen in einem Unternehmen effizienter gestalten zu können, kann ein Reservierungsmechanismus realisiert werden. Dabei soll es möglich sein, schon vor dem tatsächlichen Gebrauch der Ressource deren Nutzung planbar zu machen. Dafür ist es notwendig, die zukünftige Verfügbarkeit von Ressourcen in Erfahrung bringen zu können.

- *Spezialisierbare Darstellungen von Ressourcenlisten*

Da vom System die Definition und Modifikation von Ressourcentypen unterstützt werden soll, ist auch deren spezialisierbare Repräsentation vorstellbar. Ein Beispiel für dieses Kriterium ist der Ressourcentyp „Raum“, für den eine Darstellung des Grundrisses gewählt wird.

- *Verwendbarkeit der verwalteten Daten in Office-Anwendungen*

Hierbei sollen Schnittstellen zu bestehenden Office-Anwendungen wie zum Beispiel Microsoft Excel oder Microsoft Word geschaffen werden. Da Unternehmen zum Teil bereits Ressourcen mit solchen Applikationen verwalten, kann ihnen so die Möglichkeit einer Datenübernahme zur initialen Befüllung des

Systems gegeben werden. Des weiteren ist z.B. die Verwendung der MRMS-Daten zur Erstellung von Serienbriefen oder zum automatischen Ausfüllen von bereits eingesetzten elektronischen Formularen denkbar.

- *Passwortschutz*

Da in Unternehmen oftmals mit sensiblen Daten umgegangen wird, ist die Benutzerverwaltung mit einem Passwortschutz zu realisieren. Zudem muss für diese Passwörter ein Mindestumfang an Sicherheit gewährleistet werden. Zum Beispiel können diese eine Mindestlänge an einzelnen Zeichen haben. Ebenfalls ist eine Formatvorgabe denkbar (mindestens eine Zahl und ein Sonderzeichen).

2.3. Abgrenzungskriterien

Es ist keine Anbindung an Warenwirtschaftssysteme oder Finanzverwaltungssoftware vorgesehen. Das MRMS wird also z.B. nicht zur Lagerverwaltung geeignet sein. Es ist auch kein Content Management System, da es weder die dafür notwendigen Publizierungs- noch Freigabemechanismen unterstützen wird.

Weiterhin wird keine verschlüsselte Kommunikation zwischen Client und Server realisiert. Die Notwendigkeit ein solches Kriteriums ist nicht gegeben, da für sämtlichen Datenaustausch die Infrastruktur eines bereits abgesicherten Intranets vorausgesetzt wird.

Ein Mechanismus zum Update während des laufenden Betriebs wird nicht implementiert, siehe Abschnitt 5.3 [7].

3. Funktionalität

3.1. Muss-Funktionen

Neben den Verwaltungsfunktionen (Anlegen, Bearbeiten, Löschen) für die Benutzer, Ressourcentypen und Ressourcen sind zurzeit folgende Muss-Funktionen zu identifizieren:

- *Gefilterte Liste von Ressourcen ausgeben*

Hierbei sollen die im System verwalteten Ressourcen nach verschiedenen Gesichtspunkten gefiltert aufgelistet werden können. Dabei soll eine Filterung zum Einen nach Ressourcentypen und zum Anderen nach Status von konkreten Ressourcen möglich sein. Ein Beispiel dafür ist eine Aufstellung aller freien Ressourcen eines bestimmten Ressourcentyps.

- *Eine Ressource an eine andere binden*

Eine der Kernfunktionalitäten des Systems ist die Bindung von Ressourcen aneinander. Dabei muss der Benutzer eine vordefinierte Bindungsregel anwenden, mit der die Ressourcen verknüpft werden.

- *Eine Ressource von einer anderen entbinden*

Damit der Ressourcenbestand und Abhängigkeiten an die Veränderungen der Arbeits - und Organisationssituation angepasst werden können, ist neben dem Binden von Ressourcen auch die Möglichkeit des Lösches solcher Bindungen zu realisieren.

- *Bindungsregeln definieren*

Hierbei definiert der Benutzer des Systems Regeln, mit denen Ressourcen potenziell verknüpft werden. Diese Regeln dienen der Konsistenzhaltung des gesamten Datenbestands bei etwaiger Änderung eines Datums oder mehrerer Daten. Solch eine Änderung ist zum Beispiel das Löschen oder Umorganisieren einer Ressource. Anhand der Bindungsregeln kann nun entschieden werden, wie mit den restlichen Ressourcen, die an die zu Löschende gebunden sind, verfahren wird bzw. wie und ob diese Änderung an weitere Ressourcen propagiert werden soll.

- *Benutzer am System anmelden*
- *Benutzer vom System abmelden*

3.2. Kann-Funktionen

Funktionalitäten aus Kann-Kriterium "Reservieren von Ressourcen":

- *Ressource reservieren*
Es kann die Möglichkeit bereitgestellt werden, eine Ressource für einen bestimmten Zeitraum schon im Voraus belegen zu können. Damit ist der Ressourcenbestand insgesamt effizienter nutzbar.
- *Liste der in einem bestimmten Zeitraum zur Verfügung stehenden Ressourcen eines Typs ausgeben*
- *Liste der Zeiträume ausgeben, in denen beliebige Ressourcen eines bestimmten Ressourcentyps zur Verfügung stehen*
- *Liste der Zeiträume ausgeben, in denen eine bestimmte Ressource zu Verfügung steht*

Funktionalitäten aus Kann-Kriterium "Unterstützung von Prozessen":

- *Prozesse mit ihren Prozessschritten definieren*
Um Prozesse verwalten zu können, müssen diese zunächst vom Benutzer definiert werden. Dabei wird eine Abfolge von einzelnen Arbeitsschritten zum Erreichen eines bestimmten Arbeitszieles festgelegt, die später ausgeführt wird. Solche Arbeitsschritte können zum Beispiel das Aufrufen einer weiteren Systemfunktion, das Informieren eines Benutzers sein, aber auch Tätigkeiten, die sich nicht direkt auf das System beziehen.
- *Liste der im Kontext möglichen Prozesse ausgeben*
- *Einen Prozessschritt als durchgeführt markieren*
- *Liste der gerade aktiven Prozesse zu einer bestimmten Ressource ausgeben*
- *Liste aller aktiven Prozesse ausgeben, bei der der zurzeit zu erledigende Prozessschritt von dem gerade angemeldeten Benutzer durchgeführt werden kann*

Funktionalitäten aus weiteren Kann-Kriterien:

- *Spezielle Repräsentation eines Ressourcentyps definieren*
- *Exportieren einer Ressourcenliste in ein von einer Officeanwendung lesbares Format*
- *Import einer Menge von Ressourcen eines Typs aus einem Exportformat einer Officeanwendung*

4. Nichtfunktionale Anforderungen

4.1. Leistungen

Das System soll sich bei einer Anzahl von bis zu 20 eingeloggten Benutzern in seinem Antwortverhalten nicht merklich verlangsamen. Durchgängig soll eine flüssige, den Operationen angemessene Arbeitsgeschwindigkeit gewährleistet werden. Wir streben eine maximale Antwortzeit von zwei Sekunden für alle Funktionen an. Für einfache Verwaltungsfunktionen werden wir diese Anforderung sehr wahrscheinlich erfüllen können; bezüglich der Antwortzeit von komplexen Funktionen (wie zum Beispiel Suchen über eine große Menge von Daten)

müssen wir zunächst evaluieren, wie sich die verwendeten Technologien, insbesondere die XML-Datenbank, verhalten.

4.2. Benutzungsoberfläche

Die Benutzungsoberfläche ist so auszulegen, dass alle Funktionen des Systems vollständig unterstützt und alle anfallenden Aufgaben erfüllt werden können; weiterhin soll sie schnell und intuitiv zu bedienen sein. Darüber hinaus soll diese ergonomisch gestaltet werden:

- *Revidierbarkeit der Eingabe*

Nahezu alle Dialoge sollen abbrechbar und Dialogschritte revidierbar sein.

- *Feedback über Systemaktivitäten*

Der Benutzer soll stets Rückmeldung darüber erhalten, welche Aktivitäten das System durchführt. So muss ersichtlich sein, ob eine Eingabe oder das Anstoßen einer Aktivität erfolgreich abgeschlossen werden konnte. Dauern Systemaktivitäten länger als 15 Sekunden, so ist eine Fortschrittsanzeige sowie eine Abbruchmöglichkeit zu realisieren.

- *Abfangen fehlerhafter Eingaben*

Fehlerhafte Eingaben des Benutzers sind abzufangen um ein daraus resultierendes unkontrolliertes Systemverhalten zu vermeiden.

- *Einheitlichkeit*

Die Oberfläche soll möglichst einheitlich zu bedienen sein. Das heißt, dass formularübergreifend identische Funktionen grundsätzlich auf die gleiche Art und Weise zugreifbar sind.

- *Geringe Komplexität*

Es ist darauf zu achten, dass bei Interaktionsprozessen seitens des Benutzers wenige Arbeitsschritte durchzuführen und geringe Wege mit der Maus zurückzulegen sind. D.h. zum Beispiel auch, dass der Aufwand zur Erreichung einer Funktion des Systems im Verhältnis zu der Häufigkeit ihres Gebrauchs steht. Weiterhin darf die Oberfläche nicht überladen erscheinen.

- *Intuitive Benutzung*

Es sollen eindeutige, verständliche Begriffe und Bezeichnungen verwendet werden. Genauso muss die Struktur der Oberfläche im Kontext von Verwaltungssoftware weitestgehend selbsterklärend sein.

4.3. Qualitätsziele

- *Geringer Änderungsaufwand*

Das Design des Systems ist so auszulegen, dass eine Änderung beziehungsweise Erweiterung weitestgehend unkompliziert mit einem möglichst geringen Arbeitsaufwand zu bewerkstelligen ist. So sollen die einzelnen Komponenten (Klient, Fachlogik, Persistenzschicht) relativ leicht ausgetauscht werden können. Klar definierte Schnittstellen im gesamten System stellen eine weitere Voraussetzung zur Erfüllung dieses Kriteriums dar.

- *Kein Wartungsaufwand*

Befindet sich das System nach der Installation und Konfiguration im laufenden Betrieb, sollte keines seiner Komponenten reguläre Wartungsarbeiten erfordern.

4.4. Abgrenzung

Das MRMS wird im Rahmen des Kurses auf Grund unzureichender Zeit und fehlender Alpha- und Betatests nicht zur Einsatzfähigkeit als Produktivsystem gebracht. Da das System unter Verwendung von Technologien implementiert wird, die uns noch nicht vollends bekannt sind, kann es sein, dass sich für uns unvorhersehbare Schwierigkeiten ergeben. So muss festgehalten werden, dass keine hohe Verfügbarkeit und keine leichte Installation garantiert werden können. Zudem kann eine Portierung des Systems auf andere Plattformen - als die unter den Betriebsbedingungen genannten - nicht gewährleistet werden.

5. Einsatz

5.1. Anwendungsbereiche

Das zu entwickelnde System soll in allen Organisationsstrukturen angewendet werden können, in denen Ressourcen alltäglicher Prozesse verwaltet werden. Hierbei kann es sich auf der einen Seite um statische, nicht bewegliche Ressourcen wie zum Beispiel Räume handeln. Auf der anderen Seite soll die Verwaltung von beweglichen Gegenständen wie Computern, Telefonen oder Beamern unterstützt werden. Auch virtuelle, nicht anfassbare Dinge finden dabei ihre Berücksichtigung. Solche virtuelle Ressourcen können Software-Lizenzen oder auch Gehälter sein.

Organisationsstrukturen sind hierbei Vereine, Unternehmen und Abteilungen, die einen Personalumfang von ca. 30 - 500 Mitarbeitern umfassen.

5.2. Zielgruppen

Potenziell wird allen Mitarbeitern der Anwendungsbereiche des Systems die Möglichkeit geboten, an der ganzheitlichen Verwaltung von Ressourcen mitzuwirken. Dabei nehmen die Benutzer verschiedene Rollen ein, die sich auch in der Arbeitspraxis wieder finden lassen. So werden auf der einen Seite solche Verantwortlichkeitsbereiche innerhalb einer Organisationsstruktur unterstützt, die die Ressourcen bestimmen, die in den alltäglichen Arbeitsprozessen eingebunden sind. Ebenso richtet sich das System an solche Kräfte, die Ressourcen verwalten, also bestellen, freigeben, aussortieren oder in einer anderen Art und Weise mit diesen umgehen müssen.

Die Administratoren des Systems sollten Wissen bezüglich der Installation und Konfiguration von mehrschichtigen, verteilten Systemen auf Windows-Plattformen gesammelt haben. Weiterhin sollten Kenntnisse über Benutzer- und Rechteverwaltung vorhanden sein. Bei Benutzern der Anwendung wird eine Vertrautheit mit der Benutzung von Windows-Applikationen vorausgesetzt. Insbesondere sollten sie bereits ähnliche Software (Verwaltungssoftware) angewendet haben.

5.3. Betriebsbedingungen

Das System wird in einer normalen Büroumgebung eingesetzt und kann nachts heruntergefahren werden, so dass eventuelle Updates zu dieser Zeit passieren können. Weiterhin bedarf es keiner ständigen Beobachtung; es liegt als ein unbeaufsichtigter Betrieb vor.

6. Umgebung

6.1. Software

Das System kann mit allen Komponenten (Klient, Fachlogik, Persistenzschicht) auf folgenden Plattformen eingesetzt werden: Microsoft Windows NT, 2000 und XP. Soll nur der Klient ausgeführt werden, reicht auch eine der Consumer-Varianten von Microsoft Windows: 95, 98, ME. In jedem Fall muss die .NET-Runtime zur Verfügung stehen. Diese kann kostenlos aus dem Internet heruntergeladen werden; des weiteren besteht die Option, sie mit dem MRMS auszuliefern.

6.2. Hardware

Die einzelnen Software-Komponenten sollen auf üblicher Hardware laufen, die sich normalerweise bereits in den Räumen der Organisation befindet, die das zu entwickelnde System einsetzen wird. Es wird von einer

Hardwareplattform mit folgender minimaler Ausstattung ausgegangen: 1GHz Taktfrequenz des Prozessors, 128MB Arbeitsspeicher je an dem System beteiligten Rechner, 10MBit TCP/IP basiertes Netzwerk, bei dem eventuell vorhandene Firewalls so konfiguriert sind, dass sie HTTP- bzw. .NET-Remoting-Kommunikation zulassen (siehe Abschnitt 8 [8]). Für den Server, auf dem voraussichtlich sowohl die Datenbank als auch die Fachlogik laufen werden, wird von einer Speicherausstattung von mindestens 512MB ausgegangen.

6.3. Orgware

Es sind die zu verwaltenden Ressourcentypen und deren Abhängigkeiten sowie die Verantwortlichkeiten der mit dem System arbeitenden Mitarbeiter festzustellen, damit das System von einem Administrator entsprechend konfiguriert werden kann, d.h. die Metadaten entsprechend angelegt werden können. Üblicherweise stehen diese Informationen bereits fest - es hat sich im Laufe der Unternehmensentwicklung herausgestellt, welche Ressourcentypen von welchen Verantwortlichen zu verwalten sind.

7. Daten und Umfänge

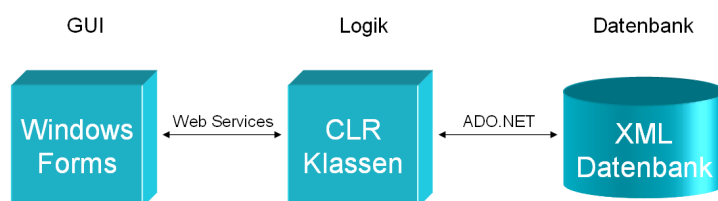
Es sollen Informationen über die Ressourcen eines Unternehmens und die Prozesse und Reservierungen, die sich auf diese Ressourcen beziehen, gespeichert werden. Als voraussichtliche Obergrenzen dienen folgende Umfänge:

- 1000 Mitarbeiter
- 10 Ressourcentypen
- 1 Ressourcen pro Mitarbeiter und Ressourcentyp, also 10000 Ressourcen
- 50 Prozesse
- 20 Prozessschritte pro Prozess, also 1000 Prozessschritte
- 100 Prozessinstanzen
- 3 Jahre Reservierungszeitraum (zwei Jahre in die Zukunft und ein Jahr Archiv)
- 10% der Ressourcentypen sind solche, die reserviert werden können, also 1000 reservierbare Ressourcen
- mit 4h durchschnittlicher Reservierungszeit, 8h Arbeitstagen und 5*52 Arbeitstage pro Jahr ergeben sich $1*5*52*3*1000$, also 1,5 Mio. zu speichernde Reservierungen

8. Einzusetzende Technologien

Die Software ist verteilt implementiert und setzt auf Microsofts .NET Plattform auf. Die Aufteilung ist die einer klassischen Drei-Schichten-Architektur: Klient, Fachlogik und Persistenz.

Abbildung 1. Drei-Schichten-Überblick



8.1. Zur Laufzeit einzusetzende Technologien

Sowohl die Fachlogik, als auch die Benutzerschnittstelle werden in der Sprache C# (C-sharp) entwickelt. Je nach Schicht kommen verschiedene .NET Bibliotheken zum Einsatz, die im Folgenden kurz beschrieben sind.

- *Benutzerschnittstelle / Klient*

Der Klient ist eine Anwendung, die auf einen entfernten MRMS-Server zugreift. Die zur Kommunikation eingesetzten Techniken und Bibliotheken werden durch die von der Fachlogik angebotene Schnittstelle (WebServices oder .NET Remoting) bestimmt. Die graphische Benutzungsoberfläche wird vollständig unter Verwendung der .NET WindowsForms Bibliothek umgesetzt.

- *Fachlogik*

Die Fachlogik muss die eigentliche Funktionalität des Systems implementieren und über eine Schnittstelle durch Klienten im Netzwerk nutzbar sein.

Der entfernte Zugriff auf die Schnittstelle erfolgt idealerweise über WebServices, wobei noch zu evaluieren ist, ob dies technisch sinnvoll ist. Einige Probleme sind z.B. niedrige Performance gegenüber nativem RPC und fehlende Unterstützung von verteilten Transaktionen. WebService-Komponenten werden von einem Webserver über HTTP angeboten. Dies übernimmt Microsoft's IIS (Internet Information Server).

Sollten WebServices sich als unzureichend erweisen, kann alternativ .NET Remoting eingesetzt werden (die RPC-Implementierung des .NET Frameworks; RPC = „Remote Procedure Call“). Kommt .NET Remoting zum Einsatz wird keine zusätzliche Software zur Kommunikation benötigt.

Die Implementierung der Fachlogik bedient sich keiner speziellen Komponenten-Technologie (wie etwa COM+), sondern läuft in der normalen .NET Laufzeitumgebung. Für die Anbindung der Fachlogik an die Datenhaltung des Systems kommt die Bibliothek ADO.NET zum Einsatz.

- *Persistenz*

In der Datenhaltung des Systems kommt idealerweise ein XML-DBMS zum Einsatz. Welches konkrete Produkt diese Aufgabe übernehmen kann, ist noch zu evaluieren. Es stellt sich auch die Frage, ob ADO.NET Zugriff auf XML-Datenbanken ermöglicht.

Sollten unvorhergesehene Probleme auftauchen, die wir nicht lösen können, so behalten wir uns vor, (auch teilweise) auf uns bekannte Technologien auszuweichen: Klient: Swing oder Struts, Fachlogik: EJB, Persistenz: ein relationales DBMS mit SQL Schnittstelle (beispielsweise MySQL oder PostgreSQL).

8.2. Entwicklungswerkzeuge und -techniken

Alle Ergebnisse, die in den verschiedenen Phasen (Analyse, Design, Implementierung, Test) der Entwicklung entstehen, werden in einem CVS-Repository verwaltet. Zu diesem Zweck ist das Projekt bei SourceForge.net angemeldet (<http://mrms.sourceforge.net>) und ist dort öffentlich zugänglich.

Die in der Analyse- und Designphase verwendeten Diagramme der UML werden mit Borland's CASE-Tool Together erstellt. Die Ergebnisse lassen sich als Bilder verschiedener Formate exportieren, die dann in die Dokumentation mit einfließen.

In der Implementierungsphase werden die Quelltexte mit VS.NET (VisualStudio.NET) erstellt und kompiliert. Wahlweise kann aber auch ein einfacher Texteditor in Kombination mit Microsoft's .NET SDK (Software-Development-Kit) verwendet werden.

Die Dokumentation erfolgt zum großen Teil unter Einsatz des DocBook-Formats (<http://www.docbook.org>). DocBook ist eine XML-Anwendung für das Verfassen von Artikeln und Büchern. Sie ist weit verbreitet und wird in vielen OpenSource-Projekten eingesetzt. Ein Vorteil: im Gegensatz zu Binärformaten wie zum Beispiel MS-Word Dokumenten, lassen sich XML-Dokumente hervorragend unter Verwendung von CVS verwalten. Zudem lassen sich aus dem Zwischenformat DocBook sehr leicht HTML- und PDF-Dokumente generieren, die dann veröffentlicht werden können. An dieser Stelle kommen die XSL-Stylesheets von Norman Walsh zum Einsatz (<http://docbook.sourceforge.net/projects/xsl/>). Sie definieren das Layout und Aussehen der Zieldokumente. Zur Generierung von PDF-Dokumenten verwenden wir den Formatting Objects Processor (FOP) der Apache Software Foundation (<http://xml.apache.org/fop/>). Die derzeitigen Ergebnisse sehen durchaus

annehmbar aus; da sich die verwendeten Tools aber noch in der Entwicklung befinden, bitten wir etwaige Formatierungsfehler zu entschuldigen. Erstellt und bearbeitet werden die DocBook Dokumente mit dem freien XML-Editor XXE (<http://www.xmlmind.com/xmleditor/>), der eine pseudo-WYSIWYG DocBook Unterstützung bietet.

Zusätzlich zu den DocBook Dokumenten werden mit MS-PowerPoint erstellte Foliensätze (*.ppt) verwendet, die ebenfalls im CVS-Repository abgelegt werden.

9. Machbarkeit

Tabelle 1. Aufwände: Analyse

Tätigkeit / Gebiet	Zeitaufwand [Mensch*Tage]
Pflichtenheft	5
Geschäftsprozesse Muss-Kriterien	5
Geschäftsprozesse Kann-Kriterien	15
Klassendiagramme Muss-Kriterien	10
Klassendiagramme Kann-Kriterien	20
UI-Prototyp Muss-Kriterien	10
UI-Prototyp Kann-Kriterien	15
<i>Summe Muss-Kriterien</i>	<i>30</i>
<i>Summe Kann-Kriterien</i>	<i>50</i>

Der Arbeitsaufwand liegt je nach Menge der zu analysierenden Anforderungen zwischen 30 und 80 Menschtagen. Diese sind durch drei Projektteilnehmer zu teilen, so dass mit einem Arbeitsaufwand zwischen 10 und 27 Tagen für jeden Teilnehmer zu rechnen ist. Für die Analysephase stehen noch 12 Wochen zur Verfügung. Rechnet man also mit einem vollen Arbeitstag pro Woche, der für das Projekt verwendet wird, so scheint der Umfang der Muss-Kriterien erfüllbar. Die Kann-Kriterien sind definitiv nicht in Gänze zu erfüllen, aber einige ausgewählte Kriterien können vielleicht in der Analyse berücksichtigt werden.

Tabelle 2. Aufwände: Design und Implementierung der Muss-Kriterien

Tätigkeit / Gebiet	Zeitaufwand [Mensch*Tage]
Rahmenwerk	Design: 5, Implementierung: 5
Verwaltung der Ressourcen in den Ressourcenlisten	Design: 5, Implementierung: 5
Definition der Ressourcentypen	Design: 5, Implementierung: 5
Benutzer- und Rechteverwaltung	Design: 5, Implementierung: 5
<i>Summe</i>	<i>40</i>

Tabelle 3. Aufwände: Design und Implementierung der Kann-Kriterien

Tätigkeit / Gebiet	Zeitaufwand [Mensch*Tage]
Modifikation der Ressourcentypen	Design: 5, Implementierung: 5
Definition von Prozessen	Design: 5, Implementierung: 5
Automatisierung von Prozessschritten	Design: 5, Implementierung: 5
Kommunikationsfunktion	Design: 5, Implementierung: 5
Reservierung von Ressourcen	Design: 5, Implementierung: 5
Spezialisierbare Darstellungen von Ressourcenlisten	Design: 5, Implementierung: 5
Verwendbarkeit der verwalteten Daten in Office-	Design: 5, Implementierung: 5

Tätigkeit / Gebiet	Zeitaufwand [Mensch*Tage]
Anwendungen	
Summe	70

Die zur Verfügung stehende Zeit von 16 Wochen für Design und Implementierung ist deutlich knapper bemessen als die für die Analyse vorgesehene. Der Arbeitsaufwand liegt grob (auf)gerundet zwischen 15 und 40 Tagen pro Projektteilnehmer. Rechnet man wieder mit einem vollen Arbeitstag in jeder der für Design und Implementierung zur Verfügung stehenden 15 Wochen, so scheint auch hier die Erfüllung der Muss-Kriterien realistisch. Wiederum mit Sicherheit nicht erfüllbar sind die Kann-Kriterien. Im Gegensatz zur Analysephase kann hier aber noch nicht einmal mit einer teilweisen Umsetzung gerechnet werden. Wir lassen sie dennoch in dieser Projektstudie stehen für den Fall, dass wir deutlich weniger Zeit als angenommen für die Umsetzung der Muss-Kriterien benötigen.

Tabelle 4. Aufwände: Einarbeitung in neue Technologien

Tätigkeit / Gebiet	Zeitaufwand [Mensch*Tage]
C# (durch jeden Projektteilnehmer durchzuführen)	$3 * 5 = 15$
XML-Datenbank evaluieren und einarbeiten	3
ADO.NET	3
Laufzeitumgebung der Logikschicht evaluieren	3
Web Services vs. .NET Remoting evaluieren	3
Summe	27

Der Aufwand für die Einarbeitung in neue Technologien liegt bei circa 30 Menschtagen, also 10 Menschtagen pro Projektteilnehmer. Da wir hierfür ein Zeitraum von 9 Wochen Semesterferien haben, erscheint dies realistisch. In dieser Tabelle sind jedoch die möglicherweise auftretenden Schwierigkeiten, die sich aus der Verwendung unbekannter Technologien ergeben, nicht berücksichtigt. Diese können zu niedrigerer Produktivität führen; rechnet man mit einem um 50 Prozent erhöhten Arbeitsaufwand, so ist die Menge der Muss-Kriterien nur erfüllbar, wenn jeder Projektteilnehmer 1,5 Tage pro Woche an Design und Implementierung arbeitet. Da wir ein hohes Interesse daran haben, uns mit den neuen Technologien zu beschäftigen, gehen wir das Risiko des möglicherweise erhöhten Arbeitsaufwandes ein.