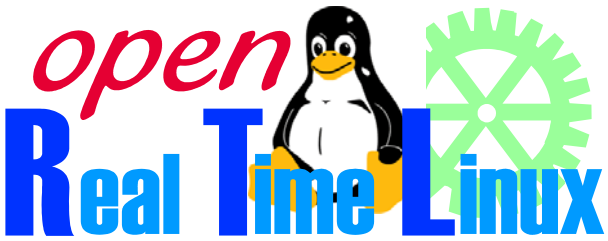
Fraunhofer Institut
Experimentelles
Software Engineering

RTLOpen Plattform Handbuch
Teil 2 – Plattform Aufbau

Autoren:
Ralf Kalmar (IESE)
Thorsten Keuler (IESE)
Stefan Meininger (Hofmann)
Dr. Klaus Viebig (VisionTools)
Holger Wußmann (Berghof)



IESE-Report Nr. 054.07/D
RTLOpen-Report 003/D
Version 1.0
1. September 2007

Eine Publikation des Fraunhofer IESE

Eine Veröffentlichung des RTLOpen-Projekts FKZ 01 IS C14

Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen FKZ 01 IS C14 gefördert.

Das Fraunhofer IESE ist ein Institut der Fraunhofer-Gesellschaft.
Das Institut transferiert innovative Software-Entwicklungstechniken, -Methoden und -Werkzeuge in die industrielle Praxis. Es hilft Unternehmen, bedarfsgerechte Software-Kompetenzen aufzubauen und eine wettbewerbsfähige Marktposition zu erlangen.

Das Fraunhofer IESE steht unter der Leitung von
Prof. Dr. Dieter Rombach (geschäftsführend)
Prof. Dr. Peter Liggesmeyer
Fraunhofer-Platz 1
67663 Kaiserslautern

Abstract

Das Dokument der RTLOpen Plattform Architektur beschreibt den Aufbau der Plattform zur Erstellung eingebetteter Echtzeit-Systeme im Maschinenbau. Sie soll dabei allen die Software in Verbindung mit Hardware produzieren die wichtigsten zu berücksichtigenden Aspekte aufzeigen und als Entscheidungshilfe im Systementwurf dienen.

Schlagworte: reference architecture , system partitioning, platform architecture, real-time system, RTLOpen

Inhaltsverzeichnis

1 Einleitung

Dieses Dokument ist Teil 2 des RTLOpen Plattform Handbuchs.

Die RTLOpen Plattform Dokumentation:

Teil 1: Übersicht

Teil 2: Plattform Architektur (dieses Dokument)

Teil 3: Softwareentwicklungsmethode für Echtzeitsysteme

Teil 4: Echtzeit-Linux

Teil 5: Entwicklungswerkzeuge

Teil 6: Plattform Demonstrator und Beispiel

Zusätzliche Information findet man auf der Projekt-Homepage:

www.open-realtime-linux.de

Das Projekt RTLOpen (Open Realtime Linux für den Maschinenbau) entwickelt eine Referenzplattform die flexibel, hoch performant und zukunftssicher ist. Sie basiert auf Open-Source Komponenten wie beispielsweise dem Betriebssystem Linux. Die RTLOpen Plattform ist in einem Handbuch dokumentiert das aus verschiedenen Teilen besteht. (s. oben)

Das Projekt ist Teil der Deutschen Forschungsinitiative "Software Engineering 2006" (SE-2006) des Bundesministeriums für Bildung und Forschung. Die Projektnummer ist 01 IS C 14.

1.1 Motivation und Hintergrund

Der ursprüngliche Ansatz des RTLOpen-Projekts, eine gemeinsame Software-Architektur zu entwickeln und zu verwenden, hat sich aus mehreren Gründen nicht als nicht praktikabel erwiesen:

1.) Die Anforderungen im Maschinenbau sind höchst unterschiedlich – unterschiedliche Hardwareplattformen, Leistungsklassen und Funktionsanforderungen lassen sich nicht mit ein und derselben Architektur unterstützen.

2.) Es gibt nicht „das" Linux. Jede Echtzeit-Linux-Lösung basiert auf meist handverlesenen Komponenten, da bisher noch kein Distributor existiert, der die

Anforderungen des Maschinen- und Anlagenbaus zufriedenstellend gelöst hat.

Aus diesem Grund wurden im Projekt einzelne Lösungsstrategien entwickelt, im Kontext einer Entwicklungsplattform strukturiert und im RTLOpen Handbuch dokumentiert, so dass individuelle Projekte leicht aufgesetzt werden können.

Die *RTLOpen-Plattform* besteht als Ganzes gesehen aus logischen Partitionen von Softwarebausteinen (z.B. Betriebssystem, IDE, usw.), eine die Entwicklung unterstützende Toolkette, sowie einer darunter liegende Entwicklungsmethode.

Dieses Dokument gibt den Überblick zu den einzelnen Elementen dieser Plattform, die in eigenen Teilen der Plattform Dokumentation näher beschrieben sind.

1.2 Wer sollte dieses Dokument lesen?

Das Dokument eignet sich als Einstieg in die RTLOpen Plattform Dokumentation, da es alle wichtigen Teile kurz und prägnant beschreibt.

Dieses Dokument ist interessant für Entwickler, die die vorgestellte Plattform inklusive Werkzeuge nutzen möchten. Des Weiteren ist dieses Dokument für Projektleiter nützlich, die sich einen Überblick über die adressierte Domäne und die Werkzeuge verschaffen möchten um Potential für eigene Projekte daraus ableiten zu können.

1.3 Struktur und Konzept des Dokuments

Zunächst werden die Begriffe Plattform und Architektur definiert. Basierend auf diesen Begriffen wird ein präzises Verständnis des Inhalts dieses Dokuments möglich.

Im zweiten Teil wird dann die RTLOpen Plattform Architektur im Einzelnen vorgestellt und im Detail auf die Einzelbausteine eingegangen. Die Plattform im Kontext des RTLOpen Projekts umfasst die komplette Entwicklungsumgebung (IDE), integriertes Projektmanagement für interdisziplinäre Systeme, eine durchgängige Methodik zur systematischen Softwareentwicklung und als auch ein durchgängiges Beispiel an dem die wesentlichen Aspekte konkret nachvollzogen werden können. Das Dokument der Plattformarchitektur wird dabei im Wesentlichen auf die Konzepte und deren Zusammenspiel eingehen und verzichtet dabei explizit auf einen überhöhten Detaillierungsgrad. Die entsprechenden Abschnitte können bei Bedarf genauer in den entsprechenden Dokumenten des Plattform Handbuchs nachgelesen werden.

Weiterführende Literatur und Referenzen befinden sich im Anhang.

1.4 Definitionen

Plattform

Eine Plattform ist eine integrierte und organisierte Menge gemeinsamer Eigenschaften auf deren Basis eine Menge von Produkten oder eine Produktfamilie erstellt werden kann.

Konzeptionell gesehen ist eine Plattform eine Relationsbeziehung. So ist Java beispielsweise eine Plattform für Objektorientierung, oder EJB („Enterprise Java Beans") eine Plattform für Java. In der hier betrachteten Domäne spielt Linux mit RTAI die Rolle der Plattform für alle Anwendungen, die in der Anwendungsschicht ausgeführt werden.

Architektur

Bei Architekturen unterscheidet man in der klassischen Informatik zunächst zwischen Systemarchitektur und Softwarearchitektur.

Bei der *System*-Architektur handelt es sich um die einem Computer auf Hardware-Ebene zugrunde liegenden Ressourcen, auf welche über das Betriebssystem, evtl. auch indirekt über ein Netzwerk, zugegriffen werden kann, sowie um die grundlegenden Eigenschaft und die Organisationsstruktur dieser Ressourcen.

Die *Software*-Architektur eines Programms oder Computer Systems ist die Struktur (oder die Strukturen) eines Systems bestehend aus Software Komponenten inklusive deren extern sichtbaren Eigenschaften und Beziehungen untereinander.

Im Falle der Plattform-Architektur betrachten wir die Bestandteile und deren Organisation die zur Herstellung der oben beschriebenen Produkte bezüglich einer Anwendungsdomäne (hier: echtzeitfähige, eingebettete Systeme im Maschinenbau) nötig sind.

2 Architektur Übersicht

In der Plattform Architektur liegt der Fokus auf der Methode der integrativen Anwendungsentwicklung. Die Methode unterstützt eine systematische Entwicklung von echtzeitfähigen Systemen basierend auf einem Linux Kernel der durch die Erweiterung RTAI („Real-time Application Interface“) [RTAI] Echtzeitfähigkeit erlangt. Zudem wird durch die Integration mit der ProMiS-Methode („Projektmanagement interdisziplinärer Systeme“) [ProMIS] die Schnittstelle zur Hardwareentwicklung explizit. Begleitet von einer Werkzeugkette, die an den Aktivitäten der Softwareentwicklungsmethode ausgerichtet ist, werden bestimmte Prozesse systematisch unterstützt. Durch eine Demonstratoranwendung schließlich wird beispielhaft die Anwendungsmöglichkeit der Plattform vorgeführt.

In Abbildung 2-1 ist die RTLOpen Plattform Architektur dargestellt. Die rot markierten Felder beschreiben Aspekte, die der Nutzer über Software beeinflusst.

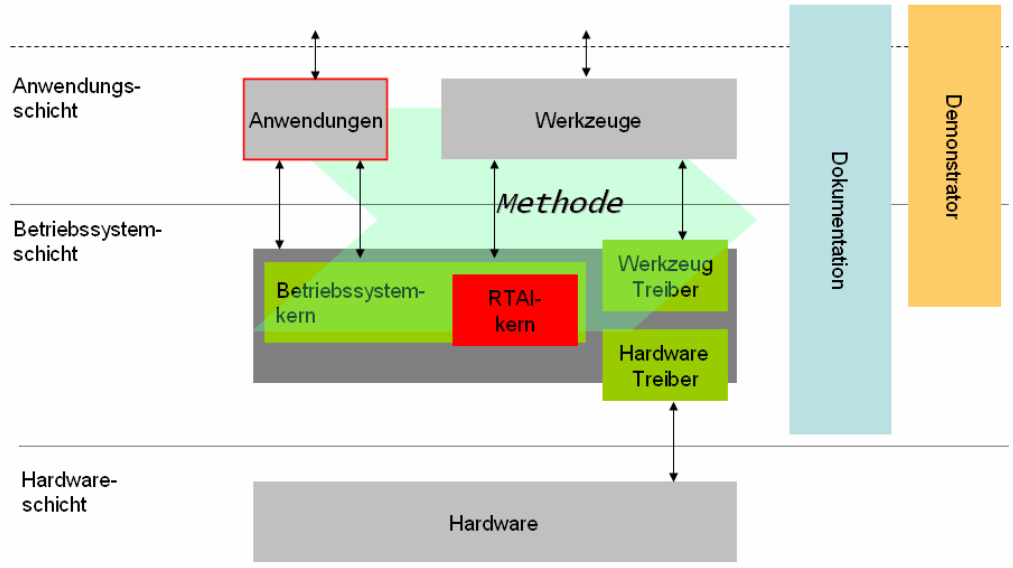


Abbildung 2-1: RTLOpen Plattform Architektur

2.1 Anwendungsschicht

Die Anwendungsschicht enthält alle Systembestandteile die in Software realisiert sind und nicht in der Betriebssystemschicht laufen. Wie bereits in Kapitel

1.4 beschrieben, betrachten wir aus Sicht der Anwendungsentwicklung das Betriebssystem Linux mit RTAI als Plattform.

Der Zugriff auf die darunter liegende Betriebssystemschicht erfolgt über definierte Schnittstellen. (s. Betriebssystemschicht)

2.1.1 Anwendungen

Anwendungen stehen im Mittelpunkt der Betrachtungen der Softwareentwicklung. Hier befinden sich die Softwarearchitektur sowie Softwareartefakte die auf Basis der Systemanforderungen und der daraus abgeleiteten Softwareanforderungen erstellt wurden. Grundsätzlich wird jedes Gesamtsystem welches auf einer explizit definierten Softwarearchitektur basiert, als Anwendung betrachtet.
Das Methodendokument als Teil 3 der Gesamtdokumentation geht im Detail auf die Prozesse und die Produkte im Softwareentwicklungszyklus ein.
Die Nutzung der Schnittstellen zum Zugriff auf die Betriebssystemschicht wird im Teil 4 der Gesamtdokumentation erläutert.

2.1.2 Werkzeuge

Im Kontext der Plattform-Architektur werden Werkzeuge zur Unterstützung der Softwareentwicklung (wie Compiler, Debugger, usw.) vorgeschlagen. Ein separates Dokument welches im Detail auf die eingesetzten oder einsetzbaren Werkzeuge eingeht und eine beispielhafte Toolkette definiert, findet man in Teil 5 der Gesamtdokumentation.

2.2 Betriebssystemschicht

Die Betriebssystemschicht stellt Dienste für die Anwendungen zur Verfügung, um auf Hardware-Ressourcen zuzugreifen. Das hier betrachtete Betriebssystem (Linux mit RTAI-Erweiterung) ist verantwortlich für die Verwaltung der Hardware-Ressourcen und die Organisation von Prozessen der Anwendung, insbesondere der zeitkritischen Tasks, die nach harter Echtzeit verlangen. Das Betriebssystem bietet die Möglichkeit der Integration verschiedenster Hardware, die über Treiber softwareseitig angesprochen werden können. Zusätzliche Gerätetreiber in der Betriebssystemschicht können weitere Dienste anbieten, wie z.B. die Ansteuerung von Aktuatoren oder das Abfragen von Sensordaten.

2.2.1 Hardwaretreiber

Hardwaretreiber sind in diesem Zusammenhang dafür verantwortlich eine klare Schnittstelle zu angeschlossenen Geräten bereitzustellen, um diese aus Softwaresicht zugreifbar zu machen. Im Hinblick auf Echtzeitfähigkeit müssen in der Regel spezielle Treiber eigens für die RTAI-Erweiterung erstellt werden. Diese Aspekte werden im Detail in Teil 4 der Gesamtdokumentation aufgezeigt.

Beispiele für Hardwaretreiber:

- Kameratreiber
- Stellmotortreiber
- CAN-Bus-Treiber

2.2.2 Werkzeugtreiber

Werkzeugtreiber sind notwendig, um Entwicklungswerkzeuge plattformspezifisch anzupassen und einsetzen zu können. Aufgrund der typischen Trennung von Entwicklungs- und Zielumgebung müssen Entwicklungswerkzeuge wie beispielsweise Echtzeit-Debugger im Entwicklungssystem lauffähig sein. Werkzeugtreiber werden üblicherweise bei Software-Werkzeugen mitgeliefert und müssen in der Regel nicht selbst angepasst oder entwickelt werden.

2.2.3 Real Time Application Interface (RTAI)

Die RTAI-Erweiterung stellt eine Reihe von Diensten für die Spezifikation von Echtzeitprozessen, sowie deren Verwaltung und Kommunikation zur Verfügung.

2.2.3.1 LXRT Dienste

LXRT (Linux-Realtime) ist ein RTAI-Modul welches es ermöglicht RTAI Dienste aus der Anwendungsschicht heraus zu nutzen.

Die Funktionalität wird mittels RTAI Prozessen umgesetzt, die dann die jeweiligen LXRT Prozesse der aufrufenden Funktion ausführen. Immer wenn ein LXRT Prozess eine Funktion der RTAI API aufruft werden die Funktionsparameter in den Adressraum des RTAI Prozesses kopiert. Es ist möglich sowohl jegliche Systemaufrufe (Linux) als auch jeden RTAI Dienst von einem LXRT Prozess aus aufzurufen.

LXRT User-space Prozesse interagieren mit RTAI Prozessen über die gleiche API (Gemeinsamer Speicher, Send-Nachrichten, Semaphore, usw.).

Soft real time in user space API	
rt_task_init	rt_sem_init
rt_mbx_init	rt_get_adr
rt_register	rt_get_name
rt_drg_on_adr	rt_drg_on_name
rt_allow_nonroot_hrt	

Tabelle 1 Schnittstelle des LXRT

2.2.3.2 Erweiterte LXRT Dienste

Die Erweiterung der LXRT Dienste betrifft in diesem Modul die Echtzeitfähigkeit der ausgeführten Prozesse. Durch Aufruf der Funktion `rt_make_hard_real_time()` wird der jeweilige Prozess einer noch höheren Priorität zugeordnet als bei herkömmlichen LXRT Prozessen.

Der Prozess wird dabei vom Linux Kern mittels „bottom-half" gescheduled. („bottom-half" beschreibt ein internes Linux Systemprogramm welches Interrupt-Dienste ausführen kann).

Hard real time user space API
rt_make hard_real_time
rt_make_soft_real_time

Tabelle 2 Schnittstelle des erweiterten LXRT

Anforderungen an Linux sowie im speziellen an Echtzeitfähigkeit im Kontext der Domänenbetrachtungen findet man detailliert in Teil 4 der Gesamtdokumentation.

2.3 Hardwareschicht

Die Hardwareschicht beschreibt alle Geräte die durch das System beeinflusst werden oder die das System beeinflussen. Das Betriebssystem ist für eine korrekte Abbildung softwareseitiger Funktionen auf die physikalischen Schnittstellen zur angeschlossenen Hardware verantwortlich. Prinzipiell verbergen sich unter der Hardwareschicht alle Geräte die über Software ansprechbar sein müssen. (Beispielsweise Sensoren, Aktuatoren oder auch der Prozessor).

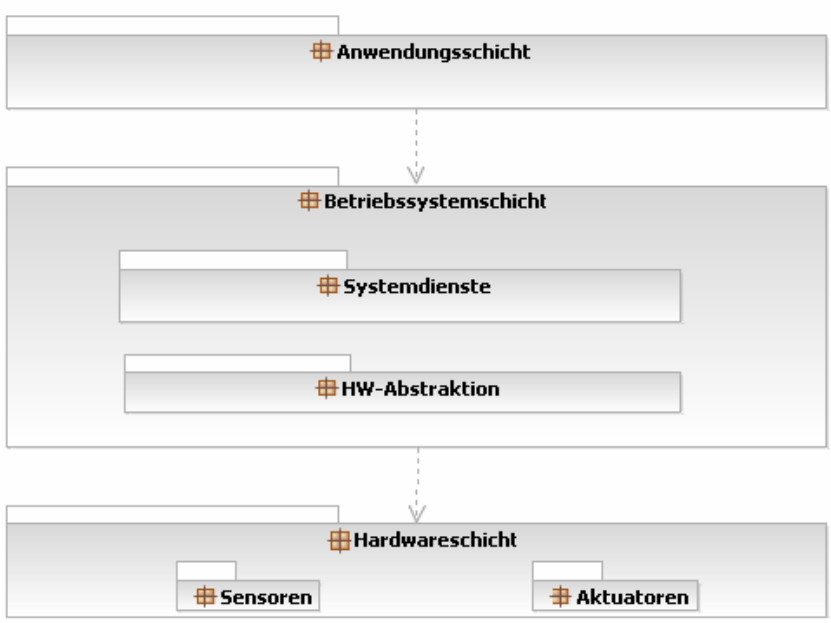


Abbildung 2 Hardwareabstraktion durch den Betriebssystemkern

2.4 Methode

Die Methode konzentriert sich auf eine integrative Entwicklung von Software und Hardware basierend auf Systemanforderungen in der Domäne des Maschinenbaus. Sie ist insbesondere dafür verantwortlich eine systematische und durchgängige Vorgehensweise aufzuzeigen die nicht nur in einer korrekten und validen Umsetzung der Spezifikation mündet, sondern die auch bereits erstellte Software systematisch wieder verwendbar und wartbar werden lässt. Hauptbestandteile der Methode sind dabei systematische Anforderungserhebung von Echtzeitsystemen sowie Architekturkonzepte wie Muster und Stile für Echtzeitsysteme.

Die Entwicklungsmethode wird im Teil 3 der Gesamtdokumentation detailliert erläutert.

2.5 Dokumentation und Demonstrator

Die RTLOpen-Plattform ist umfassend dokumentiert. Die Aspekte „Methode“, „Werkzeuge“, „Betriebssystem“ sind in eigenen Dokumenten erläutert (s. Einleitung am Anfang dieses Berichts). Eine beispielhafte Anwendung der RTLOpen-Plattform ist im Rahmen des RTLOpen-Projektes mit einem Demonstrator erfolgt. Dieser im Rahmen des RTLOpen-Handbuchs ebenfalls vollständig dokumentiert.

Zusätzlich zu diesem Demonstrator wurden industrielle Fallstudien durchgeführt.

2.6 Einschränkungen

Die Plattform-Architektur wie hier dargestellt ist jedoch in erster Linie zugeschnitten auf den Kontext der Entwicklung von Echtzeitsystemen basierend auf Linux/RTAI und der gleichzeitigen Entwicklung von Hardware, bzw. deren Integration durch Anforderungsabbildung auf Software bzw. Hardware.

Die Plattform-Architektur ist jedoch im Hinblick auf Hardware oder Betriebssysteme durchaus übertragbar auf beispielsweise Multiprozessorsysteme, verteilte Systeme oder sogar heterogene Systemintegrationen.

3 Beispiele aus der Anwendung

Die betrachtete Domäne ist der Maschinenbau im eingebetteten Bereich mit Anforderungen an Echtzeitdatenverarbeitung. Da der Ansatz eingebettet ist in ein Integrationsmodell welches zur parallelen Entwicklung von Hardware und Software dient, können zumindest modulare Aspekte des Gesamtansatzes als allgemein einsetzbar verstanden werden. So ist beispielsweise der Softwareentwicklungsstrang auch losgelöst vom konkreten Kontext der Hardware/Software-Co-Entwicklung ein gültiger und effektiver Ansatz der in andere Domänen mit softwareintensiven Systemen durchaus übertragen werden kann.

3.1 Hofmann Maschinen

Bei Hofmann sollte eine Reifengleichförmigkeits-Messmaschine (RGM) mit einem Realtime-Linux Messkern ausgestattet werden. Dazu wird eine Felge mit montiertem Reifen rotiert. Ein Lastrad wird mit einer definierten Kraft gegen das rotierende Rad gefahren. Dieses ist in Kraftmessdosen aufgehangen, mit Hilfe derer die auf den Reifen wirkenden Kräfte erfasst werden. Der Reifendruck, die ungleiche Materialverteilung im Innern der Reifenkarkasse und auf der Reifenoberfläche bewirken einen Kraftverlauf. Aus diesem lassen sich Aussagen über Laufverhalten und Qualität des Reifens ableiten.

Die Datenerfassung und Auswertung erfolgt auf einem PC auf dem RTAI-Linux installiert ist. Die Visualisierung erfolgt auf einem zweiten PC auf dem Windows (NT, 2000, XP) installiert ist. Beide Rechner sind über Ethernet gekoppelt. Weiterhin besteht eine Kopplung zu einer S7-SPS über CAN-Bus.

Die Plattformnutzung im Kontext der Firma Hofmann sieht folgendermaßen aus:

Der Messkern läuft vollständig im „Kernel Space“ ab, ist also in der Betriebssystemschicht implementiert. Zur Speicherung der Messwerte werden FIFO’s, also separate Kernelmodule eingesetzt, um die Echtzeitanforderungen jederzeit zu gewährleisten. Die Kommunikation mit den Applikationen im Userraum wird über die Schnittstelle LXRT mit Hilfe von Shared Memory abgewickelt. Ein Beispiel dafür ist der eigens angepasste TCP-Server.

Zurzeit laufen keine Applikationen in der Anwendungsschicht. Die Trennung von Messrechner und Visualisierung ist dadurch erzwungen, dass die Diagnose der S7-SPS (PDIAG) noch nicht unter Linux verfügbar ist. Daher wird zurzeit noch ein separater Windows PC benötigt.

Sobald die Diagnose unter Linux Verfügbar ist, wird es eine Verschmelzung von Mess- und Visualisierungsrechner geben. Die entsprechenden Anwendungen laufen dann in der Anwendungsschicht (User Space) ab und nutzen die LXRT-Schnittstelle.

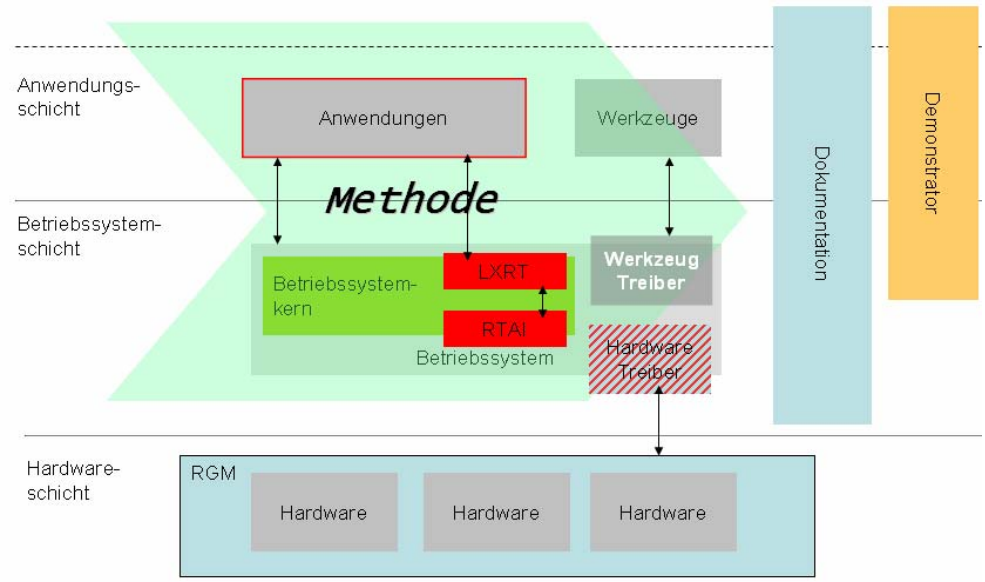


Abbildung 3-1 Plattform-Nutzung bei Hofmann-Maschinen

3.2 Berghof Automation GmbH

Die Steuerungsplattform CANtrol basierte auf 68K-Prozessoren (Motorola) und auf dem Betriebssystem VRTX. Die primären Entwicklungen für das System fanden Mitte der 90er Jahre statt. Seither wurde das System kontinuierlich weiterentwickelt und gepflegt, stößt zwischen-zeitlich aber an technische Grenzen. Problematisch dabei ist, dass Motorola keinen klassischen Nachfolger für den 68K-Baustein anbietet und auch das System VRTX abgekündigt ist. Es bot sich daher an, statt eines klassischen Re-Designs im Sinne einer Modernisierung der Architektur eine grundsätzliche Umgestaltung des CANtrol-Systems auf Basis offener konzeptioneller Ansätze durchzuführen.

Die Plattformnutzung im Kontext der Firma Berghof sieht üblicherweise folgendermaßen aus:

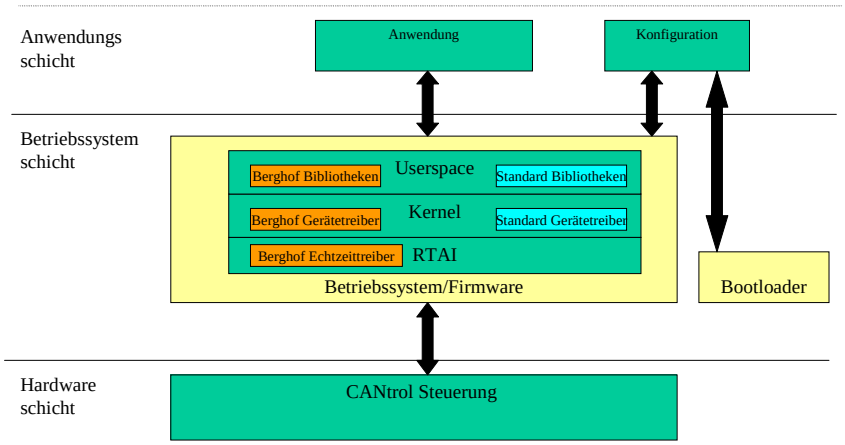


Abbildung 3-2 Plattform-Nutzung bei Berghof Automation GmbH

3.3 Vision Tools

Ziel von VisionTools ist, die auf proprietären Betriebssystemen basierenden einfachen bis mittelkomplexen Systeme und deren Vielfältigkeit (Hardware-plattformen und Softwaretools) weich auf eine Linux-Plattform zu migrieren zu können. Dabei geht es nicht um ein einzelnes Produkt, sondern um eine skalierbare universelle Plattform auf Basis von OpenSource.

Zur Evaluierung der Plattform wurde eine -Hochgeschwindig-keitskamera-anwendung mit Firewire-Kamera ausgewählt. Da man mit der aus Gründen der Wirtschaftlichkeit verwendeten Entwicklungsumgebung Kylix bei der Nutzung der Echtzeitfunktionen auf Grenzen stieß, wurden Auswerteteile in RTAI-C und Visualisierungsteile mit Kylix realisiert. Die Kommunikation zwischen dem Echt-zeitprozess und der Kylix-Visualisierung erfolgte per Shared-Memory. Dabei wurden nicht nur Ergebniskoordinaten sondern auch das komplette Kamerabild übertragen.

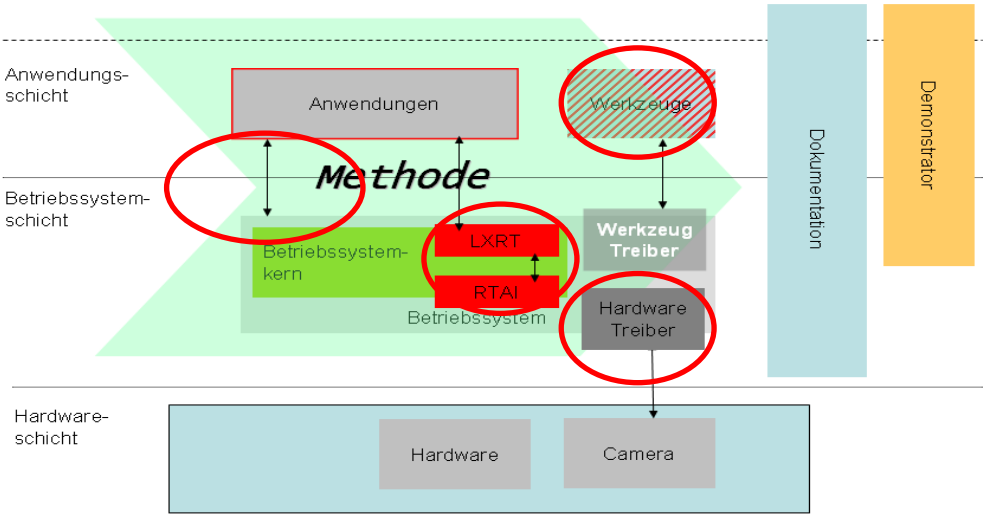


Abbildung 3 Aufspaltung von Realtime-Prozess und Visualisierungsanwendung

Ergebnis war eine Userspace-Realtime-Auswertung, die per Shared-Memory sowohl das Kamerabild, als auch Bildauswertungsergebnisse zur Verfügung stellte. Zusätzlich wurden noch Bildzähler und hochaufgelöste Zeitstempel (Nanosekunden) übertragen. Eine separate Kylixanwendung übernahm die Visualisierung und konnte mit Hilfe der Bildzähler Zeitstempel sowohl den Jitter der Zykluszeit als auch die Bildwiederholrate exakt bestimmen und anzeigen. Da die Kamera die Bilder in einem festen Takt liefert, der eventuell asynchron zum Echtzeitprozess ist, wurde ein Test- Modul im Kernelspace entwickelt, das eine Messung der Echtzeiteigenschaften ermöglichte. Dabei zeigte sich, dass der RTAI-Prozess mit minimalem Jitter (im Nanosekundenbereich) selbst bei extremen Prozessorfremdlasten lief.

Referenzen

[ProMIS] E. Geisberger, R. Schmidt: ProMiS – Projektmanagement für interdisziplinäre Systementwicklungen, VDMA, 2004

[RTAI] Realtime Application Interface, www.rtai.org

Dokumenten Information

Titel:	RTLOpen Plattform Handbuch Teil 2 – Plattform Aufbau
Datum:	1. September 2007
Report:	IESE-054.07/D
Status:	Final
Klassifikation:	Öffentlich