Christian d'Heureuse, Inventec Informatik AG, 4.1.1994

Prinzipien der Inventec-Softwaretechnologie für industrielle Beschriftungs- und Automatisierungsanwendungen

Inhalt

1. Einleitung

Dieser Text präsentiert Grundlagen und Konzepte, die bei Inventec für die Realisierung von Beschriftungs- und Automatisierungsprojekten angewendet werden. Das Schwergewicht dieses Textes liegt auf den Softwaretechniken, die im Bereich der industriellen Produktion eingesetzt werden.

1.1. Übersicht über die Softwarefunktionen

Die Softwarefunktionen bei typischen Industrie-Beschriftungsprojekten lassen sich in die zwei Bereiche "Produktion" und "Vorbereitung/Auswertung" unterteilen. Diese Funktionsbereiche sind über gemeinsame Daten verbunden.

<table>
<tr>
<td>Produktion:
Ansteuern und Überwachen der Peripheriegeräte (Industrie-Etikettendrucker, Etikettenspender, Inkjetdrucker, Barcode-Scanner, Waagen, Lichtschranken, Sensoren, Zähler, Weichen, etc.)
Automatische Abläufe
Anzeigen des Zustands des Systems
Sammeln von Produktionsdaten und Schreiben von Logeinträgen
Interface zu andern Produktionssystemen</td>
<td>Vorbereitung/Auswertung:
Erfassen und Bearbeiten von Artikel- und Auftragsdaten (SQL-Datenbank)
Erstellen von Druckerlayouts mittels Layouteditoren
Konfiguration der Software und der Peripheriegeräte
Datenbank-Pflege:
Backup/Restore (Sicherheitskopien)
Import/Export für Datenaustausch mit übergeordneten Systemen, Archivierung, etc.
Recall-System (Produkteverfolgung)
Statistische Auswertungen</td>
</tr>
<tr>
<td colspan="2">Daten:
Statische Daten (Stammdaten):
Artikeltabellen
Druckerlayoutdefinitionen
Konfigurationsdaten
Dynamische Daten:
Auftragsdaten
Logdaten</td>
</tr>
</table>

Abbildung 1: Funktionsbereiche der Software

1.2. System-Hierarchie

Beschriftungs- und Automatisierungssysteme müssen meist in übergeordnete EDV-Systeme des Anwenders integriert werden. Sie bilden die unterste Schicht eines komplexen Systemverbunds.

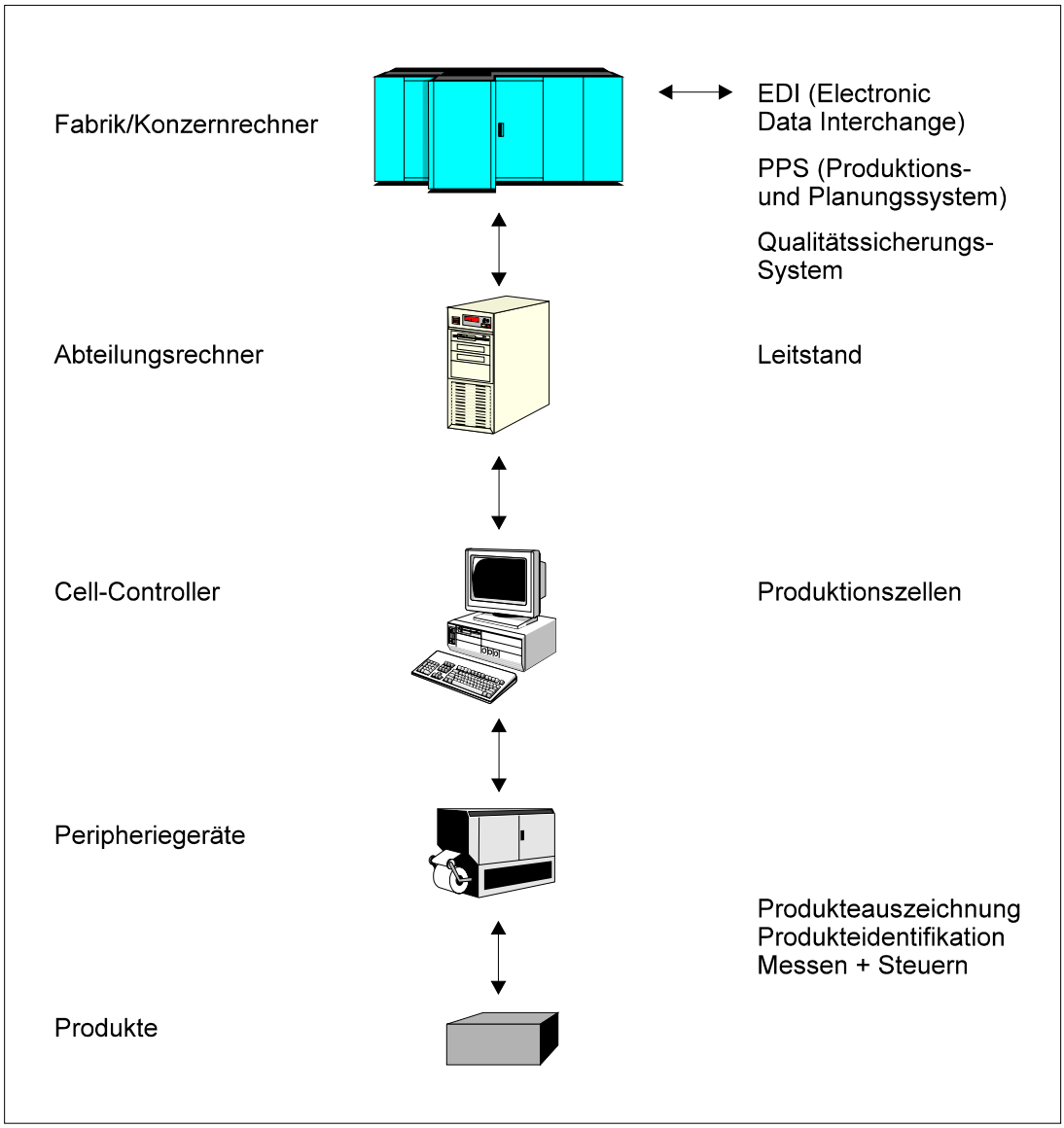

Abbildung 2: System-Hierarchie

2. Kommunikations-Architektur

2.1. RPC

RPC (Remote Procedure Call) ist eine Softwaretechnologie, die für Programm-Kommunikation nach dem Client/Server-Konzept geeignet ist. Eine RPC-Transaktion besteht immer aus einer Request-Message vom Client-Prozess zum Server-Prozess und der entsprechenden Reply-Message retour vom Server zum Client.

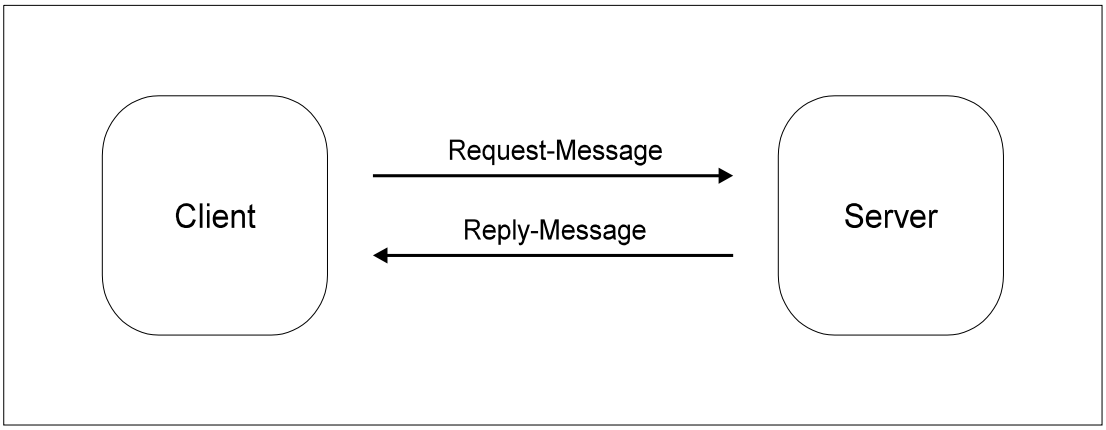

Abbildung 3: RPC

Sämtliche Inter-Prozess-Kommunikation der Inventec-Software basiert auf der RPC-Schicht. Je nach Betriebssystem und vorhandener Hardware können für das RPC-Subsystem verschiedene darunterliegende Transport-Methoden gewählt werden. RPC-Verbindungen können über LAN, über serielle Leitungen oder lokal zwischen Prozessen auf dem gleichen Computer aufgebaut werden. Die RPC-Schicht bietet nach oben eine einheitliche Schnittstelle, die unabhängig von den darunter liegenden Transport-Methoden ist. Das RPC-Subsystem enthält spezielle Funktionen für Error-Handling und Error-Recovery.

Queues	Variable Pools	Virtual Printers	Virtual Terminals	...	
RPC					
Named Pipes	NetBIOS	TCP/IP	APPC/CPI-C	Shared Memory	...
Ethernet	Tokenring	RS-232	RAM	...	

Abbildung 4: Schichten der Kommunikations-Architektur

2.2. Queues und Variable-Pools

Zur Kommunikation zwischen Programmprozessen werden die Softwareobjekte Queue und Variable-Pool verwendet.

Queues werden verwendet, um Datenrecords von einem Prozess an einen andern zu schic??ken, ohne dass der erste Prozess darauf warten muss, bis der zweite die Daten entgegengenommen hat. Die Datenrecords werden in der Queue gebuffert. Dadurch werden die zwei Prozesse entkoppelt und können asynchron zueinander laufen. Ein Realtime-Prozess kann z.B. die Daten eines gerade bearbeiteten Produkts über eine Queue einem andern Prozess zur Weiterbearbeitung übergeben.

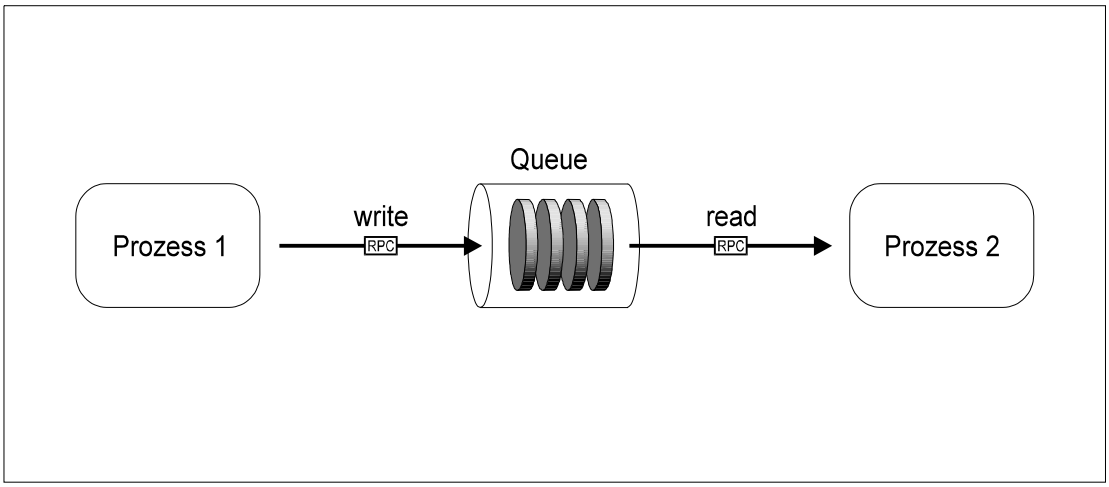

Abbildung 5: Queue

Variable-Pools werden meistens dazu verwendet, um Statusinformationen zwischen Prozessen auszutauschen.

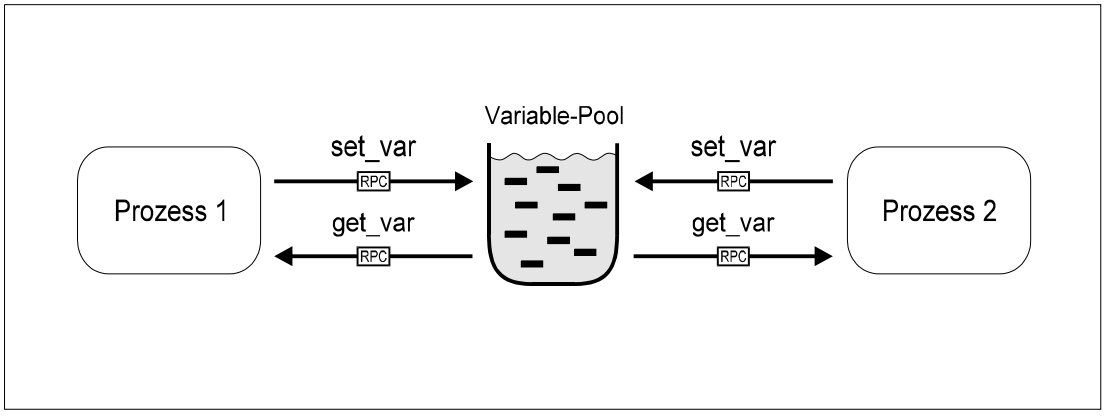

Abbildung 6: Variable-Pool

Da Queues und Variable-Pools auf der RPC-Kommunikation basieren, können sie völlig transparent auch über LAN verwendet werden. An einer Fertigungsstrasse können die

Daten über ein gerade bearbeitetes Werkstück z.B. über eine Queue von einem Computer an den nächsten weitergegeben werden. Ein Prozess, der auf einem Computer läuft, kann über LAN z.B. Statusvariablen aus einem Variablen-Pool lesen, der sich auf einem andern Computer befindet.

Die Softwareobjekte Queue und Variable-Pool sind als C++-Objekte implementiert. Es gibt von beiden je eine RPC-Client- und eine RPC-Server-Variante. Die RPC-Server-Varianten enthalten einen internen Thread, so dass sie autonom auf eintreffende RPC-Requests reagieren können. Die C++-Objekte können in DLLs verpackt auch von andern Programmiersprachen verwendet werden, z.B. von einer Makrosprache wie REXX oder von Datenbank-Frontend-Tools.

3. Virtuelle Peripheriegeräte

Virtuelle Geräte sind abstrakte Softwareobjekte, die ein reelles Gerät repräsentieren (z.B. einen Industrie-Inkjetdrucker oder eine automatische Waage). Das Konzept der virtuellen Geräte wird einerseits verwendet, um verschiedene Gerätetypen der gleichen Geräteklasse auf ein gemeinsames Interface abzubilden. Dadurch erreicht man, dass die kundenspezifischen Applikationen gerätetypenunabhängig programmiert werden können.

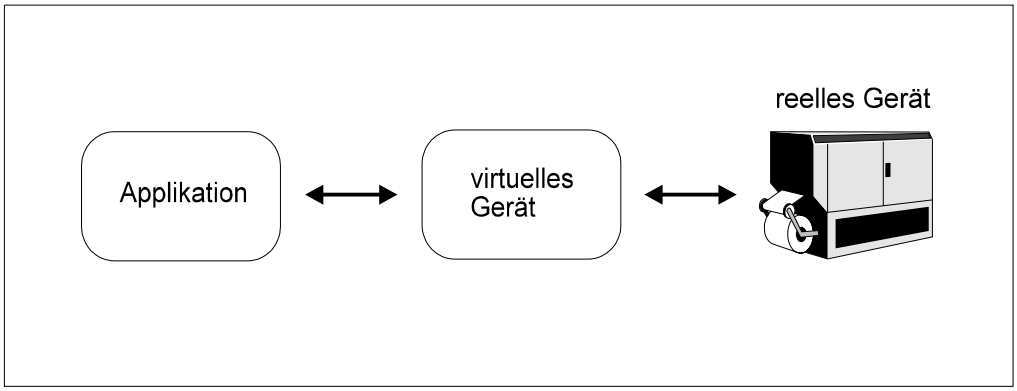

Abbildung 7: Virtuelles Gerät

Ausserdem ermöglicht es dieses Konzept, virtuelle Geräte transparent über RPC-Verbindungen zu spiegeln. Dadurch erreicht man die Ortsunabhängigkeit der an den Computern eines LANs angeschlossenen Geräte. Ein virtuelles Geräte-Objekt auf dem einen Computer kann ein reelles Gerät repräsentieren, das physikalisch an einem andern Computer im LAN angeschlossen ist. Alle Befehle, die vom Applikationsprogramm an das virtuelle Gerät geschickt werden, werden automatisch über RPC an ein zweites virtuelles Geräte-Objekt geschickt, das auf dem andern Computer ist, und gelangen von dort aus zum reellen Gerät. Umgekehrt werden z.B. Statusänderungen des Geräts automatisch vom zweiten virtuellen Geräte-Objekt an das erste zurückgeschickt.

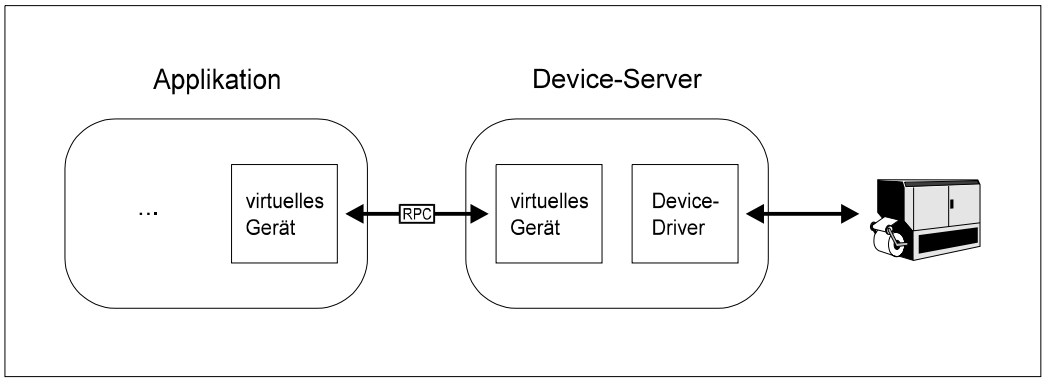

Abbildung 8: Virtuelle Geräte mit RPC-Verbindung

Wie Queues und Variable-Pools sind auch die virtuellen Geräte (virtuelle Drucker, Terminals, etc.) in C++ implementierte Softwareobjekte mit internem Thread. Sie können, in DLLs verpackt, direkt von Frontendprogrammen aus angesprochen werden.

3.1. Virtuelle Drucker

Ein virtueller Drucker repräsentiert als Softwareobjekt einen reellen Drucker, z.B. einen Etikettendrucker oder einen Industrie-Inkjetdrucker.

Bei Inkjetdruckern kann ein virtueller Drucker auch eine ganze Gruppe von Inkjet-Druckereinheiten repräsentieren. Dies ist dann sinnvoll, wenn mehrere Jet-Köpfe das gleiche Produkt beschriften. In diesem Fall wird die ganze Jet-Gruppe als logische Einheit betrachtet, der ein mehrzeiliges Layout zugewiesen werden kann. Das Layout wird dann softwaremässig auf die einzelnen Druckköpfe aufgeteilt.

Ein virtueller Drucker versteht folgende Befehle:

- open virtuellen Drucker öffnen
- close virtuellen Drucker schliessen
- print Druckertask starten
- stop Drucken stoppen
- get_status Status abfragen

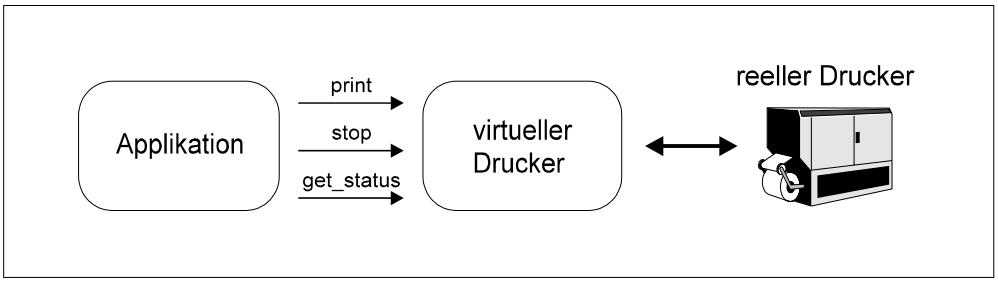

Abbildung 9: Virtueller Drucker

Da der virtuelle Drucker einen eigenen internen Thread besitzt, arbeitet er autonom und asynchron zum Applikationsprogramm. Wenn das Applikationsprogramm dem virtuellen Drucker einen Befehl schickt, muss es nicht warten, bis der Befehl an den reellen Drucker geschickt wurde, sondern kann sofort weiterarbeiten. Der virtuelle Drucker sendet Befehle selbständig zum reellen Drucker weiter und fragt periodisch den Status des reellen Druckers ab. Der virtuelle Drucker kann auch selbständig gewisse Fehlerzustände beheben, z.B. (bei gewissen Druckertypen) automatisch die Layout-Daten wieder an den Drucker schicken, wenn der Drucker aus- und wieder eingeschaltet wurde.

3.2. Drucker-Konfiguration und Layout

Bevor ein virtueller Drucker angesteuert werden kann, müssen die Konfigurations-Definition und die Layout-Definitionen erstellt worden sein. Die heute verbreiteten Industrie-Etiketten- und Inkjetdrucker haben sehr unterschiedliche Konfigurationsparameter und Layout-Definitionssprachen. Verschiedene Hersteller bieten graphische Layout-Editoren für ihre Drucker an. Die Layouts werden in einer druckerspezifischen Syntax in Files gespeichert. Manche Drucker bieten auch die Möglichkeit, die Layout-Definitionen im voraus in den Drucker zu laden.

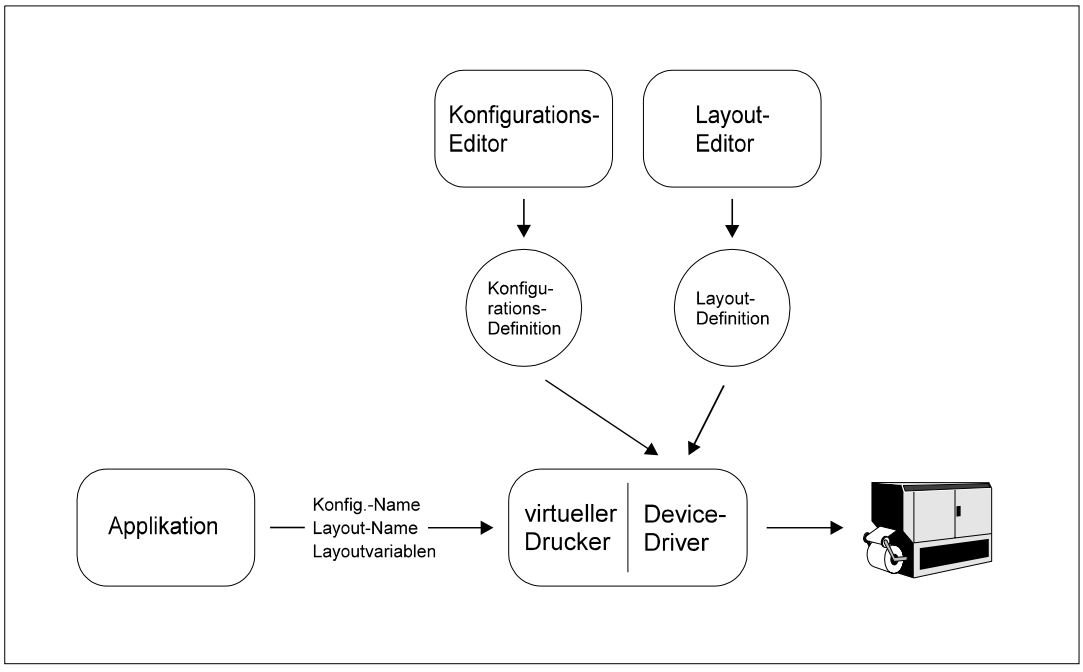

Abbildung 10: Datenfluss für virtuelle Drucker

Beim **open-**Befehl wird dem virtuellen Drucker der Name der Konfigurations-Definition übergeben. Wie die Konfigurationsdaten gespeichert sind ist, druckerspezifisch. Für Inkjet-Drucker können z.B. je nach Modell über ein Dutzend Jetkopf-Parameter eingestellt werden.

Beim **print-**Befehl wird dem virtuellen Drucker der Name der Layoutdefinition und eine Liste von Layoutvariablen übergeben. Der Devicedriver kombiniert dann die druckerspezifische Layoutdefinition mit den Werten der Layoutvariablen. Über die Layoutvariablen können z.B. variable Felder im Layout eingesetzt werden, oder es können

innerhalb des Layouts verschiedene Varianten von Sublayouts ausgewählt werden. Bei typischen Anwendungen, z.B. für die Etikettierung von Lebensmitteln, verwendet man einige Grundlayouts, in die man mittels Layoutvariablen variable Felder und Sublayouts einsetzt.

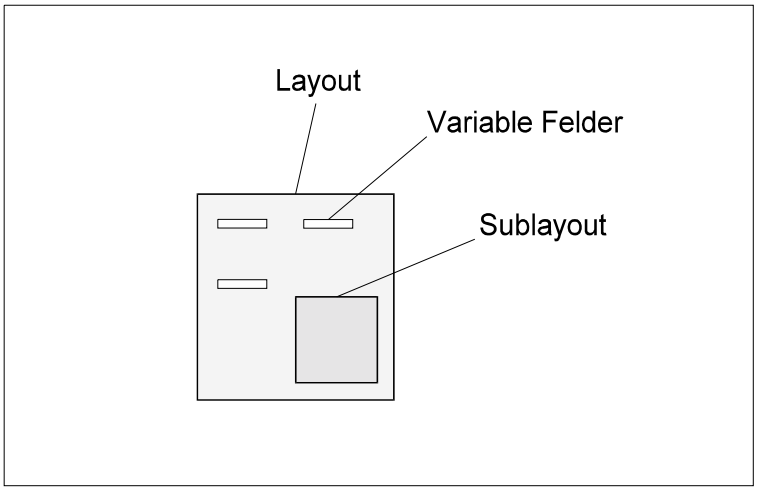

Abbildung 11: Drucker-Layout

3.3. Virtuelle Terminals

Ein virtuelles Terminal ist ein Softwareobjekt, das eine Input- und eine Output-Queue enthält und für folgende Geräteklassen verwendet wird:

- kleine Handheld-Terminals mit LCD-Display und Tastatur, eventuell kombiniert mit Laserscanner (z.B. Symbol-Scanner)
- Barcodescanner
- automatische Waagen
- Sensoren
- Input/Output über die Handshaking-Leitungen der RS-232-Schnittstelle oder die I/O-Leitungen des Parallel-Ports, z.B. für die Kontrolle eines UPS (Uniterruptible Power Supply), den Anschluss einer Alarmlampe oder andere einfache Steuer- und Messanwendungen.

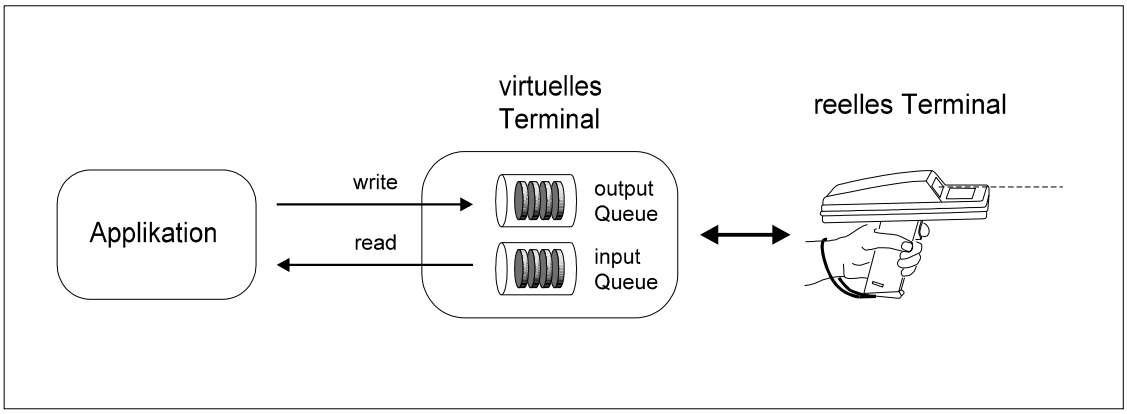

Abbildung 12: Virtuelles Terminal

4. Typische Programmstruktur

Für Softwaresysteme in der industriellen Produktion hat sich die folgende dreiteilige Programmstruktur bewährt:

- Frontendprogramm mit GUI (Graphical User Interface)
- Kontrollprogramm
- Deviceserver-Programme

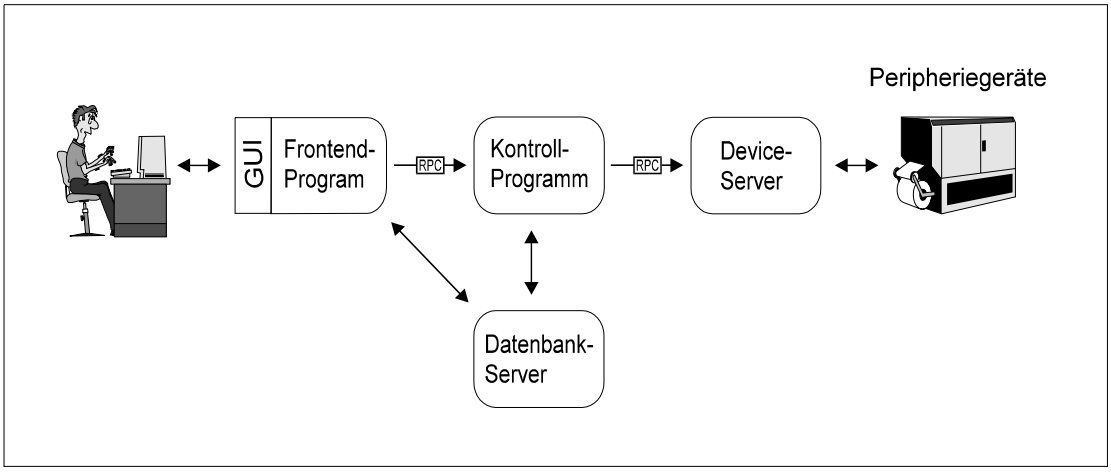

Abbildung 13: Typische Programmstruktur

Die drei Komponenten Frontendprogramm, Kontrollprogramm und Deviceserver sind über RPC verbunden. Sie können beliebig im LAN verteilt laufen. Üblicherweise laufen mehrere Instanzen der drei Programme verteilt auf den Rechnern einer Farbrikhalle. Das Frontendprogramm läuft z.B. auf Rechnern neben den Produktionsanlagen sowie im Leitstandraum. Zwischen den Programmen bestehen n-zu-n-Verbindungen. Das Frontendprogramm kann gleichzeitig mehrere Instanzen des Kontrollprogramms ansteuern, und das Kontrollprogramm steuert wiederum mehrere Deviceserver an.

Bei einfachen Anwendungen können auch zwei der Komponenten, z.B. Frontend- und Kontrollprogramm, oder auch alle drei in einem Programm realisiert werden.

Das Frontendprogramm wird mit einem Frontend-Tool für graphische Oberflächen programmiert, z.B. mit einem Visual-REXX- oder Visual-Basic-System oder mit einem Datenbank-Frontend-Tool. Es erlaubt dem Benutzer das Starten und Stoppen von Produktionsaufträgen und zeigt den Status des Systems an.

Das Kontrollprogramm steuert die automatischen Abläufe und überwacht und koordiniert die an einem Prozess beteiligten Geräte. Es verknüpft Bedienereingaben, die es vom Frontendprogramm erhält, mit Daten aus der Datenbank und leitet diese an die Deviceserver weiter. Es überwacht den Produktionsprozess und schreibt Logeinträge in die Datenbank. Im Kontrollprogramm sind die anwenderspezifischen Abläufe programmiert. Es muss für den Anwender verständlich und erweiterbar sein. Als Programmiersprache wird deshalb eine offene Standard-Scriptsprache wie REXX oder Basic verwendet. Über DLLs kann auf virtuelle Drucker, virtuelle Terminals, Queues, Variable-Pools, Semaphoren etc. und auf die Datenbank zugegriffen werden.

Die Deviceserver sind in C++ geschriebene Standardprogramme. Sie sind portabel programmiert und können deshalb auf alle modernen Betriebssysteme portiert werden. Für etliche in der Industrie verbreitete Geräte sind bereits Deviceserver vorhanden. Dank langjähriger Erfahrung und einer sorgfältig aufgebauten Softwareinfrastruktur sind die Inventec-Ingenieure in der Lage, innert kurzer Frist neue Deviceserver zu entwickeln.

5. Beispiel: Verlag Otto Maier Ravensburg, PickPack-Anlage

5.1. Übersicht

Im Sommer 1993 wurde bei der Verlag Otto Maier AG in Ravensburg ein System für das automatische Wägen und Beschriften von Paketen in Betrieb genommen. Die Anlage besteht aus zwei Teilen: Der automatischen Linie und der manuellen Linie. Die manuelle Linie ist für Pakete vorgesehen, die für die automatische Linie zu schwer oder zu gross sind. Beide Linien werden von einem PC aus gesteuert, auf dem das Betriebssystem OS/2 läuft.

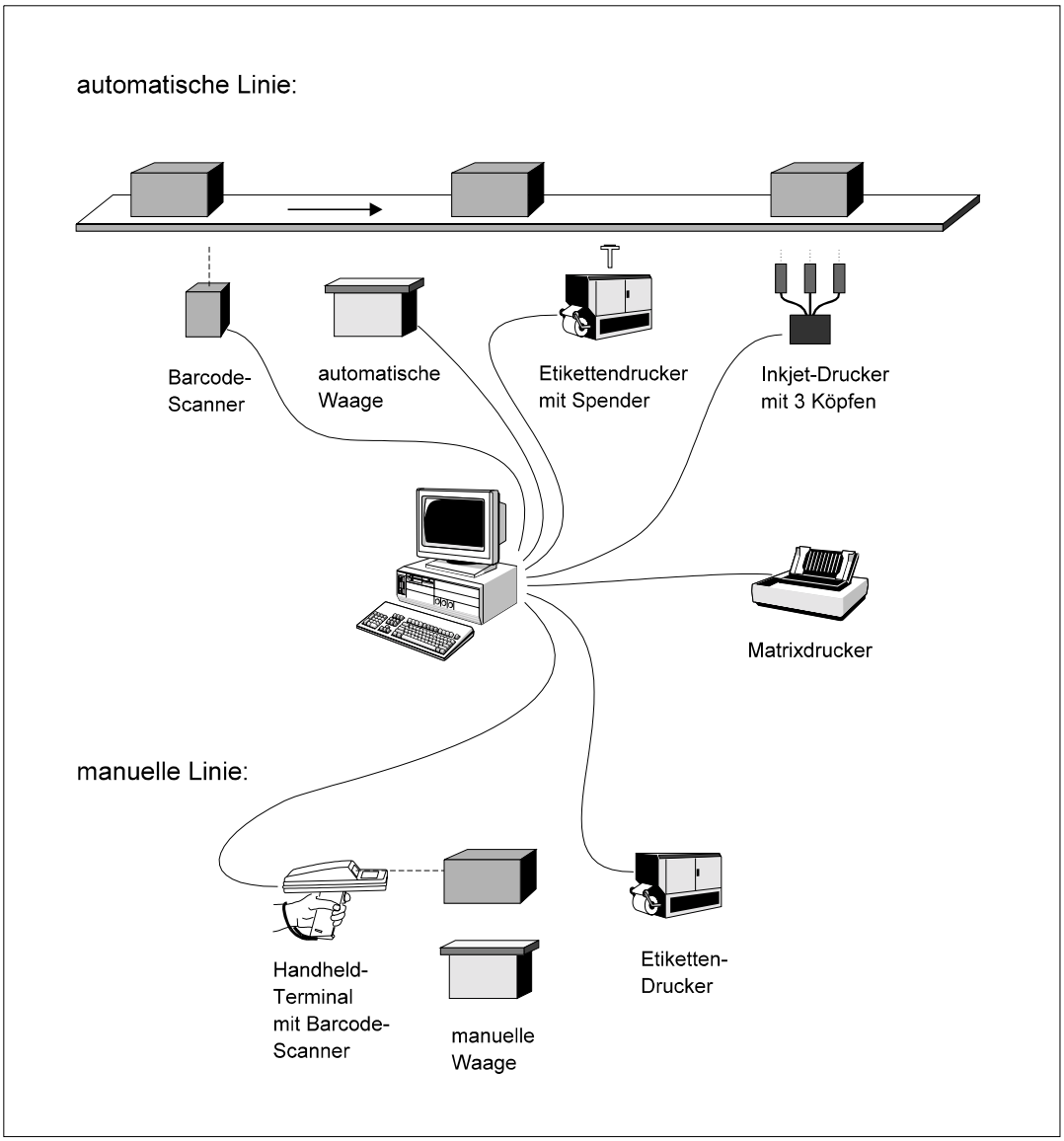

Abbildung 14: Übersicht Ravensburg PickPack-Linie

5.2. Funktionsbeschreibung

Auf der automatischen Linie kommen auf dem Fliessband Pakete mit Spielwaren mit einer Maximalgeschwindigkeit von 10 Paketen pro Minute. Die Pakete sind bereits mittels Barcode mit einer eindeutigen Paketnummer beschriftet. Zuerst wird diese Paketnummer mit dem Scanner gelesen, und das Paket wird gewogen. Das Kontrollprogramm sucht dann in der Datenbank nach dem Eintrag für das Paket, holt die dazugehörenden Auftragsdaten und trägt das Gewicht in der Datenbank ein. Dann wird eine Etikette gedruckt, die das exakte Gewicht und verschiedene auftragsbezogene Daten und Barcodes enthält. Anschliessend wird das Paket mit drei Inkjet-Köpfen beschriftet.

Das erste und das letzte Paket eines Auftrags, das die Anlage passiert, muss speziell gekennzeichnet werden. Pakete, die zu verschiedenen Aufträgen gehören, kommen wild durcheinander. Die Software muss deshalb für jedes Paket speichern, ob es schon bearbeitet wurde oder nicht. Hinter der PickPackanlage werden die Pakete manuell nach Aufträgen sortiert und auf Paletten gepackt. Sobald alle Pakete eines Auftrags erledigt sind, druckt die Software einen Frachtbrief, auf dem unter anderem alle Pakete mit Gewicht sowie das Totalgewicht verzeichnet sind.

Die Geräte an der automatischen Linie mussten aus mechanischen Gründen mit Zwischenabständen montiert werden. Auf der Strecke zwischen Scanner/Waage und Inkjet/Etikettendrucker können deshalb bis zu drei Pakete gleichzeitig unterwegs sein.

Auf der manuellen Linie werden die Pakete von Hand gewogen, und der Barcode wird mit einem Symbol-Handheld-Terminal gescannt. Das Gewicht wird von Hand auf dem Symbolterminal eingetippt. Auf dem Display des Terminals werden die Paketdaten zur Kontrolle angezeigt.

Je nach Versandart (Bahn, Post, German-Parcel, etc.) müssen für ein Paket andere Etiketten gedruckt werden. Für gewisse Versandarten müssen Pakete, die ein bestimmtes Gewicht überschreiten, anders beschriftet werden. Für German-Parcel muss am Ende des Arbeitstages eine Diskette geschrieben werden, auf der die Daten aller über German-Parcel verschickten Pakete verzeichnet sind.

Jeweils morgens vor und abends nach der Produktion werden die Auftrags- und Paketdaten zwischen dem PC und dem Hostsystem transferiert.

5.3. Programmstruktur

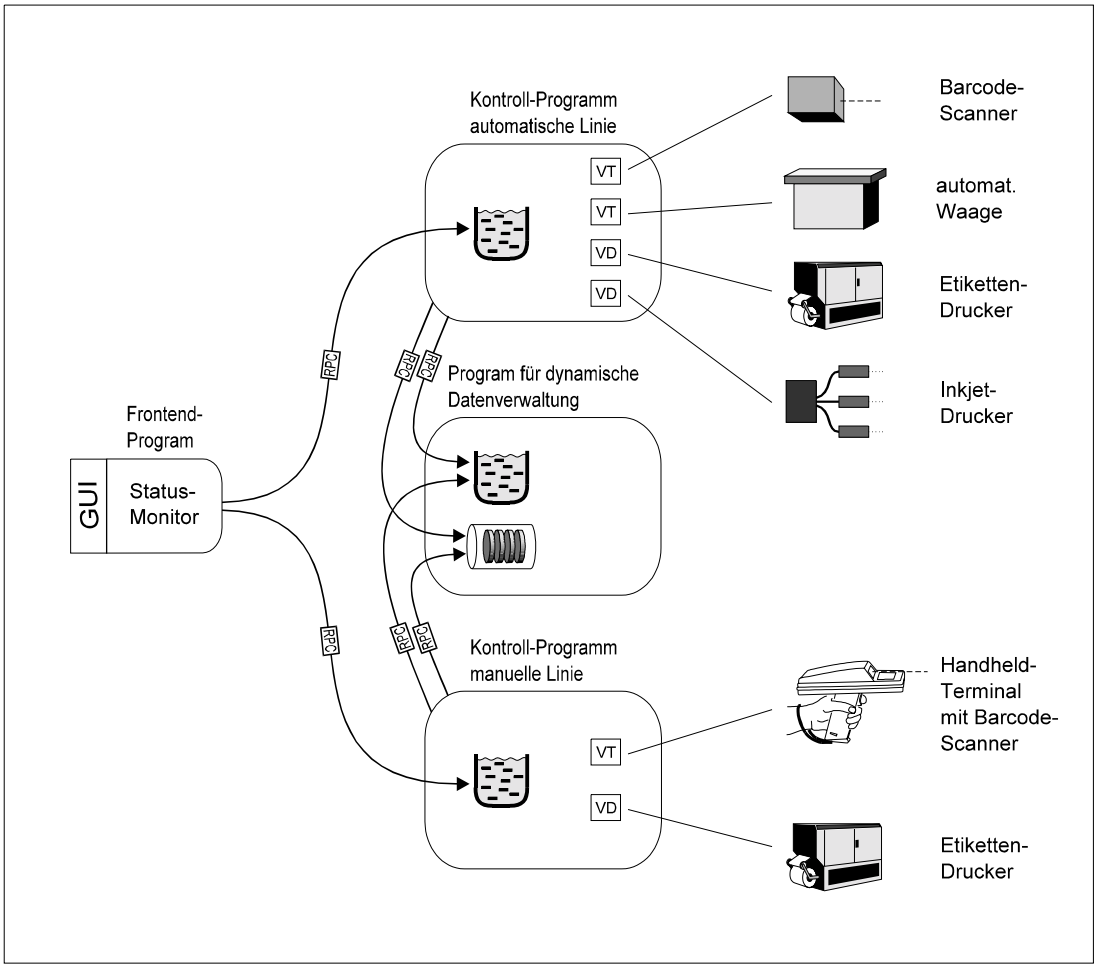

Abbildung 15: Programmstruktur Ravensburg PickPack

Die Applikationssoftware ist in folgende Programme aufgeteilt:

- Kontrollprogramm für die automatische Linie, in REXX programmiert.
- Kontrollprogramm für die manuelle Linie, in REXX programmiert.
- Hilfsprogramm für die Verwaltung der dynamischen Daten, in REXX programmiert.
- Frontendprogramm für Status-Monitor-Anzeige, in VX-REXX programmiert.

Die einzelnen Programme kommunizieren über Queues und Variable-Pools miteinander. Das Frontendprogramm holt periodisch Status-Informationen aus den Variable-Pools der beiden Kontrollprogramme und stellt den aktuellen Status auf dem Bildschirm dar.

Das Kontrollprogramm für die automatische Linie muss relativ schnell reagieren, wenn ein neues Paket eintrifft. Der SQL-Datenbankserver ist jedoch relativ träge. Deshalb werden alle benötigten Daten im RAM gebuffert. Die dynamischen Daten werden von einem separaten Hilfsprogramm in einem Variablen-Pool verwaltet. Datenänderungen, z.B. der

Wert des gemessenen Paket-Gewichts, wird von den Kontrollprogrammen über eine Queue an das Hilfsprogramm geschickt und von diesem in der Datenbank nachgeführt.

6. Beispiel: Jacobs-Suchard Bern, JasuPrint

Das Softwaresystem JasuPrint ist seit 1990 bei Jacobs-Suchard-Tobler in Bern im Einsatz. Es steuert die Beschriftung von Schokoladeverpackungen, Paketen und Paletten in der Produktion.

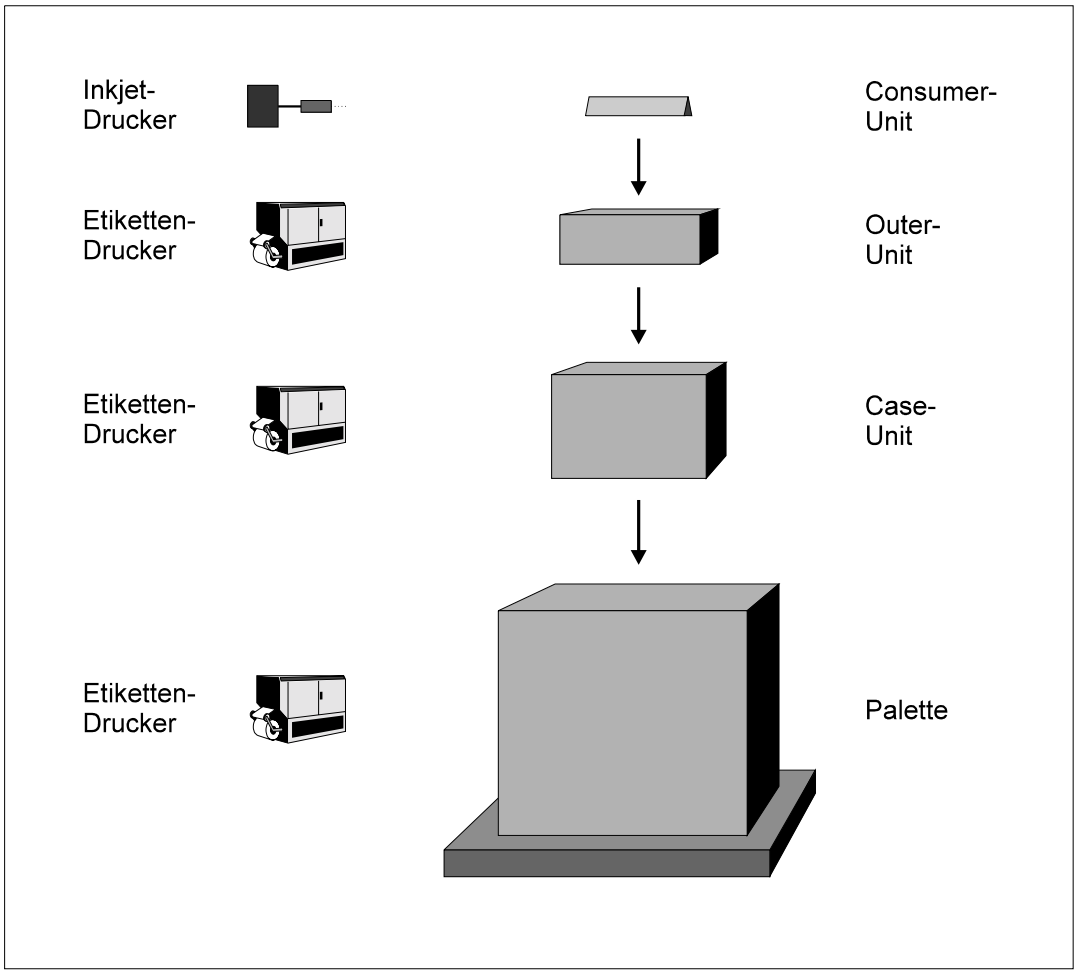

Abbildung 16: Verpackungsstufen Jacobs-Suchard JasuPrint

Auf die einzelnen Schokoladen (Consumer-Units) wird mit einem Inkjetdrucker das Verfalldatum und ein Recall-Code gedruckt. Die Pakete (Outer- und Case-Units) und Paletten werden mit Etiketten beschriftet, die je nach Bestimmungsland andere Texte und Barcodes enthalten müssen.

Sämtliche Abläufe sind automatisiert. Verfalldatum und Recall-Code werden automatisch von der Software berechnet und während der Produktion automatisch aktualisiert. Alle artikelspezifischen Informationen sind in einer zentralen SQL-Datenbank gespeichert.

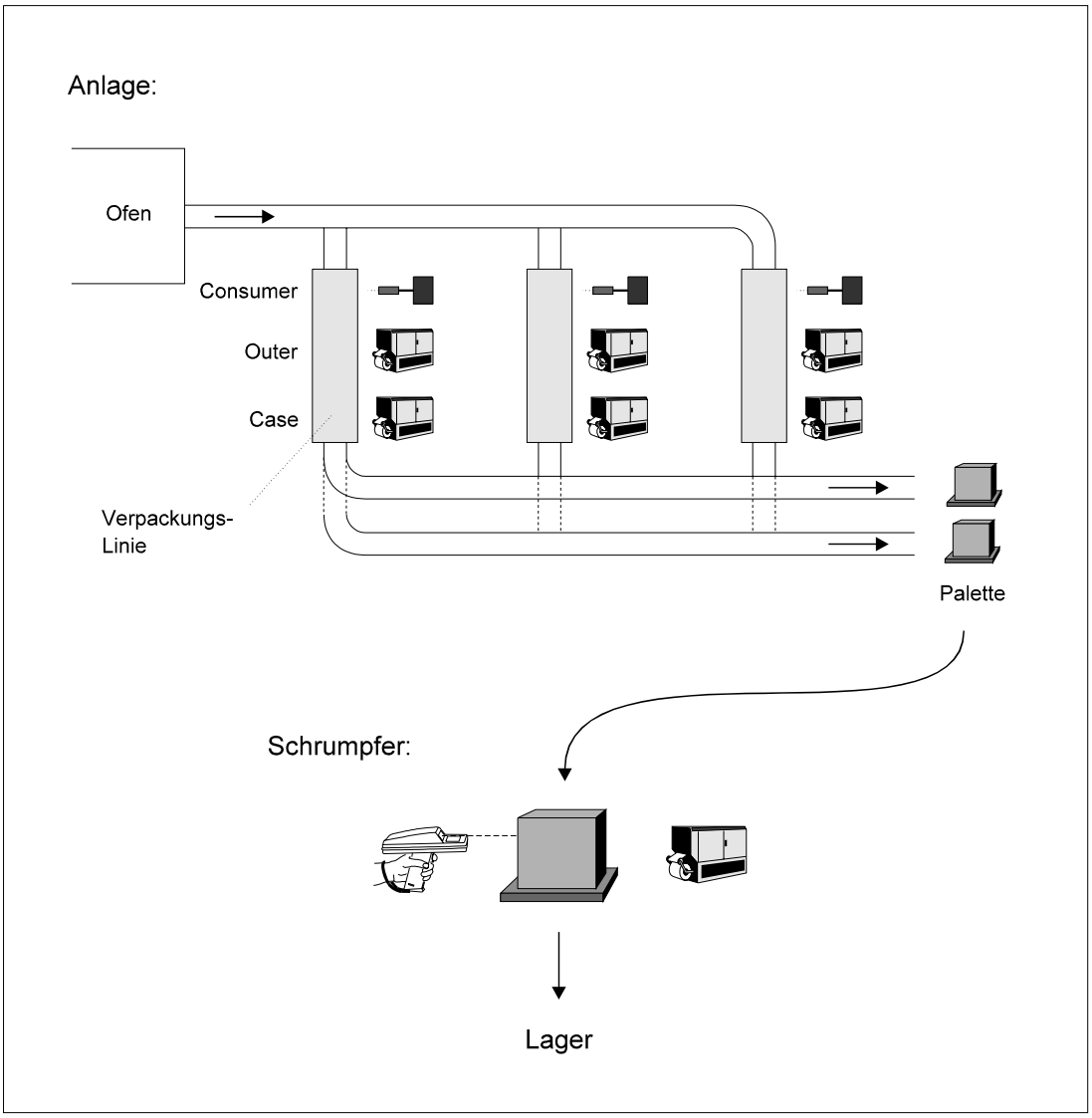

Abbildung 17: Schema einer typischen Anlage für Schokolade-Produktion

Eine typische Anlage für die Schokolade-Produktion enthält mehrere parallele Verpackungslinien. Die Schokoladetafeln, die aus dem Ofen einer Anlage kommen, werden auf den Verpackungslinien je nach Bestimmungsland verschieden verpackt und beschriftet. Softwaremässig stellt man ein, welche Linien welchen Artikel produzieren.

In einer Fabrikhalle stehen mehrere solche Anlagen. Die Paletten von verschiedenen Anlagen werden zu einem gemeinsamen Schrumpf-Platz gebracht, wo sie mit Folie umhüllt und geschrumpft werden. Mit einem Handheld-Barcodescanner wird der Recall-Code auf den Paketen gelesen, und ein Etikettendrucker druckt die entsprechende Palettenetikette.

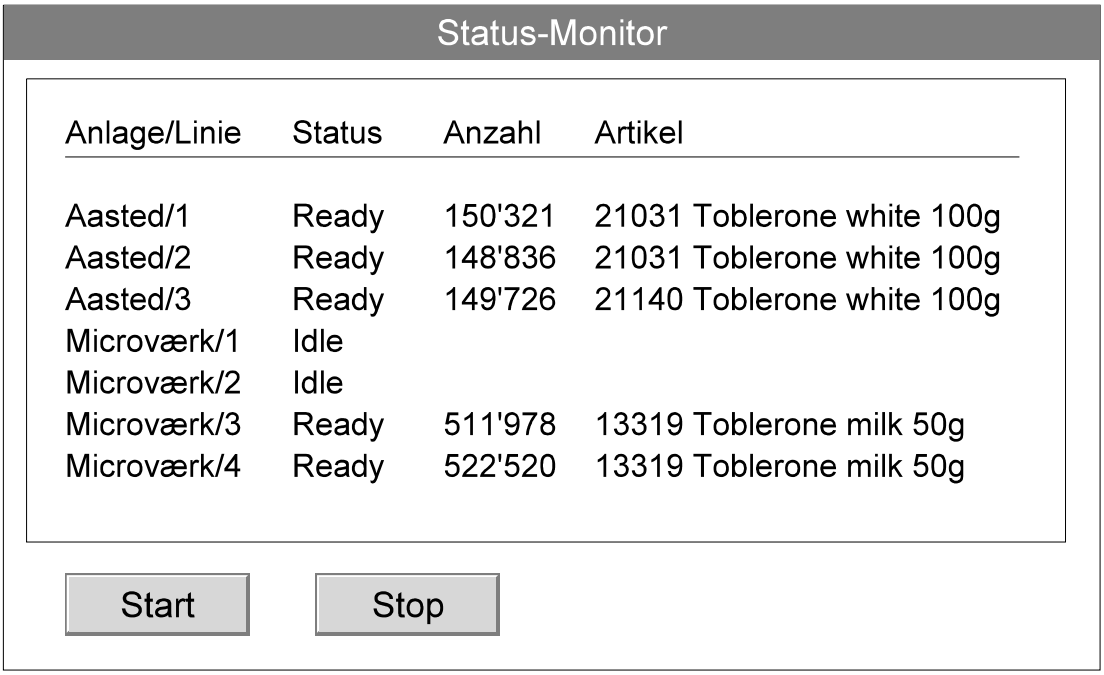

Abbildung 18: Statusmonitor-Anzeige (vereinfacht)

In den Fabrikhallen verteilt stehen mehrere OS/2-PCs, die über ein LAN verbunden sind. An den PCs sind die Etiketten- und Inkjetdrucker angeschlossen. Die Software JasuPrint läuft als verteiltes System auf diesen PCs im LAN. Von jedem PC aus kann der Status des gesamten Systems und der einzelnen Produktionsanlagen überwacht und gesteuert werden.

In der Log-Tabelle der Datenbank wird verzeichnet, welches Produkt wann wo in welcher Menge produziert wurde. Diese Daten werden für das Recall-System (Produkteverfolgung/ Qualitätssicherung) und für statistische Auswertungen verwendet.