Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 89 (1998)

Heft: 17

Artikel: Eine Plattform für Automatisierungssysteme in der Anästhesie

Autor: Frei, Christian W. / Leibundgut, Daniel / Branca, Andrea

DOI: https://doi.org/10.5169/seals-902102

Die Durchführung einer Narkose stellt dem Anästhesisten unter anderem regelungstechnische Aufgaben: Während eines chirurgischen Eingriffes muss er die Daten lebenswichtiger Körperfunktionen aufnehmen und durch Vergleiche mit entsprechenden Soll-Werten beurteilen. Abweichungen von den Soll-Werten muss der Arzt durch manuelle Regelung kompensieren. Am Inselspital Bern wird in Zusammenarbeit mit der ETH Zürich ein System entwickelt, das den Anästhesisten von einem Teil seiner Aufgaben entlasten soll, indem es ihm die eigentlichen Regelungsaufgaben abnimmt und ihn vermehrt seine Überwachungsaufgaben wahrnehmen lässt.

Eine Plattform für Automatisierungssysteme in der Anästhesie

■ Christian W. Frei, Daniel Leibundgut, Andrea Branca, Adolf H. Glattfelder und Alex M. Zbinden

Die Anästhesie hat während chirurgischer Operationen mehrere Aufgaben zu erfüllen. Es sind dies:

- Erreichen der Schmerzfreiheit
- Ausschalten von Bewusstsein und Erinnerungsvermögen
- Relaxierung der Muskeln
- Aufrechterhalten der lebenswichtigen Körperfunktionen

Um diese Ziele zu erreichen, müssen verschiedene physiologische Kenngrössen wie der Blutdruck, die Herzfrequenz, die CO_2-Konzentration in der ausgeatmeten Luft, die Sauerstoffsättigung des Blutes und andere Grössen auf bestimmten Niveaus oder in bestimmten Grenzen gehalten werden. Der Anästhesist wird von entsprechenden Geräten laufend über die aktuellen Werte dieser Grössen informiert. Er entscheidet aufgrund dieser Informationen (Ist-Werte) und der Zielvorgaben (Soll-Werte), ob und wie die Einstellungen an medizinischen Geräten wie Anästhesiegasverdampfer, Beatmungsmaschine oder Infusionspumpe geändert werden sollen. Der Anästhesist nimmt dabei ganz offensichtlich die Funktion eines Reglers wahr. Diese Routineaufgabe macht einen grossen Teil seiner Tätigkeiten aus, und es ist deshalb naheliegend, zu versuchen, ihn durch automatische Regler von dieser Routinetätigkeit zu entlasten, damit er sich ausschliesslich auf die Überwachung konzentrieren kann.

In einem Forschungsprojekt des Instituts für Anästhesiologie und Intensivmedizin des Inselspitals Bern und des Instituts für Automatik der ETH Zürich soll herausgefunden werden, in welchem Umfang automatische Regler in der Anästhesie eingesetzt werden können. Bis jetzt befasste man sich ausschliesslich mit dem Entwurf von Reglern, was vor allem die Schwerpunkte Modellierung, Identifikation, Reglerauslegung und deren Validierung in Pilotversuchen umfasste. Um diese Pilotversuche im Operationssaal durchführen zu können, wurden die Regler auf einer am Institut für Automatik entworfenen Plattform implementiert. Die Plattform basiert auf einem PC und der Programmiersprache Modula II. Sie war zunächst dazu gedacht, die Machbarkeit eines derartigen Konzeptes zu verifizieren. Entsprechend wenig Gewicht wurde auf Erweiterbarkeit, Benutzerschnittstellen-Entwurf und Wartungsfreundlichkeit gelegt. Auch die Zuverlässigkeit der Software war zweitrangig, da zur Überwachung des Reglers ein zusätzlicher Anästhesist permanent anwesend war. Nachdem nun die Mach-

Adresse der Autoren
Christian W. Frei, Andrea Branca, Prof. Dr. *Adolf H. Glattfelder,* Institut für Automatik ETH Zentrum, 8092 Zürich
Daniel Leibundgut, Prof. Dr. *Alex M. Zbinden* Institut für Anästhesiologie und Intensivmedizin Inselspital, 3010 Bern

barkeit in mehreren Studien [1, 2, 3] unter Beweis gestellt wurde, möchte man diese Regler auch routinemässig einsetzen, um zum Beispiel standardisierte Anästhesien für medizinische Studien zu erhalten. Damit verlässt man das Stadium der Machbarkeitsstudien. Gleichzeitig wird es aus Kostengründen notwendig, die Sicherheit und Zuverlässigkeit des Systems so zu erhöhen, dass es durch einen einzigen Anästhesisten überwacht werden kann. Neben dem routinemässigen Einsatz wird das Regelsystem laufend erweitert werden, indem neue Regelkreise oder neue Regeltechniken integriert werden. Daraus werden sich hohe Anforderungen an die Erweiterbarkeit des Gesamtsystems ergeben. Um diesen Bedürfnissen gerecht zu werden, werden Supervisorfunktionen implementiert. Die Erweiterbarkeit des Systems und der Benutzerschnittstellen-Entwurf erfordern den Neuentwurf der Software. Da man bereits mit dem jetzigen System an die Kapazitätsgrenzen der Hard- und Software stiess, drängte es sich auf, die gesamte Plattform von Grund auf neu zu konzipieren. Es gibt weltweit etwa ein halbes Dutzend Forschungsinstitute, die sich mit der Anwendung von Automatisierungstechnik in der Anästhesie befassen; uns ist aber nur ein Fall bekannt, wo das dazu verwendete System publiziert wurde [4], und dieses kann unseren Anforderungen nicht genügen. Es ist auch kein kommerzielles System vorhanden, welches für unsere Zwecke hätte verwendet werden können. Man war daher gezwungen, eine eigene Lösung zu suchen.

Anforderungen

Aus den Erfahrungen der ersten Machbarkeitsstudien und den Zukunftsplänen für die Forschung lassen sich folgende Anforderungen ableiten:

- maximale Sicherheit im Sinne der VDI-Definition [5]
- hohe Zuverlässigkeit und Verfügbarkeit
- einfache Erweiterbarkeit
- hohe Benutzer- und Wartungsfreundlichkeit

Diesen Anforderungen ist sowohl bei der Auswahl der Komponenten als auch beim Systemaufbau Rechnung zu tragen. Einfluss war nur bei der Wahl von Computerhardware und Software möglich. Kein Einfluss konnte auf die Auswahl von Monitoren und Aktuatoren genommen werden. Im folgenden wird nur auf die Aspekte eingegangen, die beim Entwurf auch beeinflusst werden konnten.

Sicherheit ist nach [5] «eine Sachlage, bei der das Risiko nicht grösser als das Grenzrisiko ist. Im Bereich der Automatisierungstechnik beinhaltet die Sicherheit einer Betrachtungseinheit die Fähigkeit, innerhalb vorgegebener Grenzen für eine gegebene Zeitdauer keine Gefahr zu bewirken oder eintreten zu lassen.» Nach [6] versteht man unter Risiko das Produkt aus der Wahrscheinlichkeit, mit der ein Ereignis eintritt, und dem Ausmass des daraus resultierenden Schadens, das heisst:

$$Risiko = Eintrittswahrscheinlichkeit \times Schaden \quad (1)$$

Sicherheit ist dann gegeben, wenn dieses Risiko kleiner als ein tolerierbares Grenzrisiko ist. Eine Vorgehensweise, wie mit den schwer quantifizierbaren Begriffen Eintrittswahrscheinlichkeit und Schaden für medizintechnische Anwendungen umgegangen werden kann, ist in [6] beschrieben. Gleichung (1) verdeutlicht, dass das Risiko vermindert werden kann, indem man zum Beispiel durch konstruktive Massnahmen die Eintrittswahrscheinlichkeit vermindert, also keine Gefahr wirksam werden lässt, oder indem man durch Schutzmechanismen dafür sorgt, dass gefährliche Betriebszustände erkannt werden und durch adäquate Gegenmassnahmen (z.B. durch Umschalten auf Handbetrieb) dafür sorgt, dass kein Schaden eintritt.

Die Anwendung des Sicherheitsbegriffes auf Software unterscheidet sich von der Anwendung auf Hardware in dem Sinne, dass Software keinen Abnutzungserscheinungen unterliegt, jedoch inhärent systematische Fehler (Entwurfsfehler) enthält [7]. Die in [7] aufgezeigten Paradigmen für sicherheitsgerichtete Software sind sehr streng und für ein experimentelles System wie das hier beschriebene nicht praktikabel. Es sollen aber mindestens Programmiersprache und Entwurfsverfahren genutzt werden, die den korrekten Entwurf von Software unterstützen.

Eine Entlastung für den Anästhesisten stellt das System nur dann dar, wenn es eine hohe Verfügbarkeit aufweist und zuverlässig funktioniert. Zuverlässigkeit und Verfügbarkeit werden beide von der Korrektheit der Software beeinflusst. Neben den bereits erwähnten Forderungen an Entwurfsverfahren und Programmiersprache wird an das System die Forderung nach Echtzeitfähigkeit gestellt. Im Vorgängersystem war die entsprechende Funktionalität durch den Programmierer selbst realisiert worden, was eine potentielle Fehlerquelle darstellte. Durch die Verwendung eines Echtzeitbetriebssystems wird hier die Zuverlässigkeit erhöht.

Erweiterbarkeit spielt eine weitere entscheidende Rolle. Es muss einerseits möglich sein, neue Funktionalität (z.B. Überwachung) in einen bestehenden Regelkreis einzubauen. Andererseits werden durch das Hinzufügen weiterer Regelkreise zwangsläufig weitere Messgeräte und Steuergeräte ins System integriert werden müssen. Nicht zuletzt kommt es immer wieder vor, dass der Wunsch aufkommt, Geräte durch neuere zu ersetzen, weil sie bessere Eigenschaften (z.B. eine höhere Genauigkeit) haben. All dies muss möglich sein, ohne grundlegende Änderungen am System vornehmen zu müssen. Diese Forderung stellt unmittelbare Bedingungen an Hardware und Softwarestruktur.

Da von nun an das System routinemässig durch Anästhesisten verwendet werden soll, muss die Benutzerschnittstelle

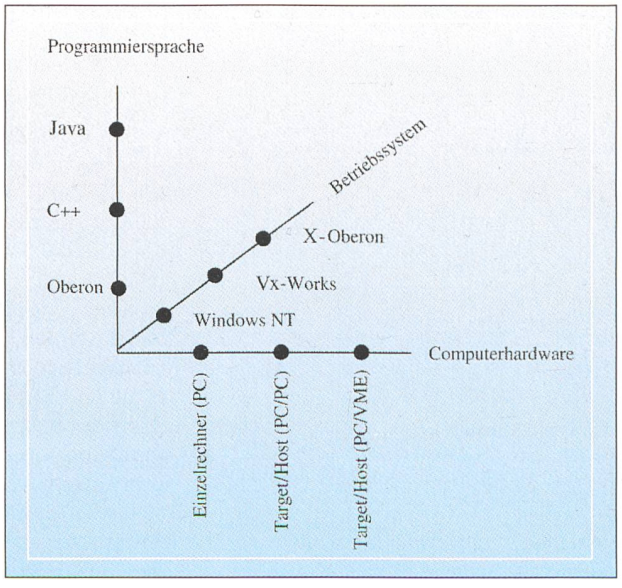


Bild 1 Die Plattform besteht aus der Computerhardware, dem Betriebssystem und einer Programmiersprache. Verschiedene Plattformen lassen sich demnach als Punkte in einem dreidimensionalen Raum auffassen.

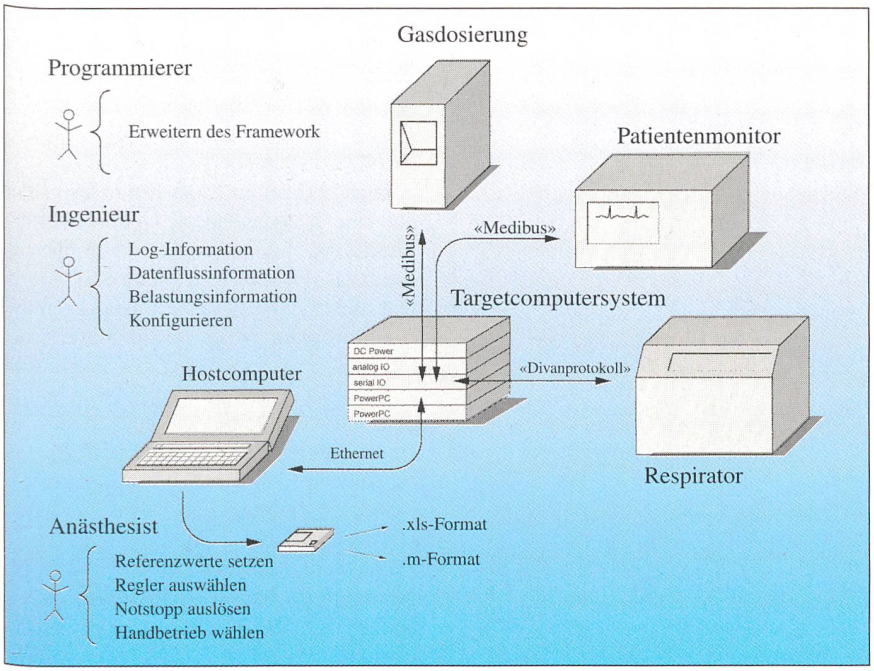


Bild 2 Context Picture
Es vermittelt einen Überblick des Systems und erlaubt die Diskussion der Anforderungen an die Software.

in einer Weise gestaltet werden können, wie man es von kommerziellen Anwenderprogrammen gewohnt ist.

Plattform

Die Plattform wird nach Bild 1 durch die drei Komponenten Computerhardware, Betriebssystem und Programmiersprache gebildet. Die Plattform bildet das Fundament für das experimentelle System. Ein solides Fundament ist nach unserer Auffassung eine Grundvoraussetzung für den Bau eines sicheren Systems, denn es macht wenig Sinn, Schutzmechanismen zu entwerfen, um Schwachstellen in der Plattform zu entschärfen, wenn dies durch eine geeignete Plattformwahl umgangen werden könnte.

Aus den Anforderungen im vorhergehenden Abschnitt wurden folgende Auswahlkriterien für die Plattform abgeleitet:

- erweiterbare Computerhardware
- echtzeitfähiges Betriebssystem
- objektorientierte Programmiersprache mit Klassenbibliothek zur Gestaltung von Benutzerschnittstellen

In einem Auswahlverfahren, in dem die einzelnen Komponenten detailliert diskutiert wurden [8], entschied man sich unter Berücksichtigung der spezifischen Randbedingungen für dieses Entwicklungsprojekt für das X-Oberon-System des Instituts für Robotik der ETH Zürich. In den folgenden Abschnitten werden die einzelnen Plattformkomponenten kurz vorgestellt.

Computerhardware

Die Computerhardware besteht aus zwei Teilen, einem Target und einem Host (Bild 2). Das Targetcomputersystem wird durch ein VME-Bus-System mit Power-PC gebildet. Auf diesem Targetsystem läuft die Ein-/Ausgabe, die Regelung und die Überwachung unter dem Echtzeitbetriebssystem X-Oberon. Die Modularität des VME-Systems stellt in komfortabler Weise die Hardwareerweiterbarkeit sicher. In einem ausschliesslich auf PCs beruhenden System wäre die Ausbaufähigkeit der Hardware bedeutend geringer. Der Hostcomputer ist ein Standard-PC mit dem Standard-Oberon als Betriebssystem, der einerseits als Entwicklungsumgebung dient und andererseits die Systemüberwachung durch den Anästhesisten erlaubt. Für die Programmierung und Systemkonfiguration wird wie üblich mit Maus und Tastatur gearbeitet, die Bedienung während des Betriebs erfolgt dann allerdings über einen Touchscreen.

Betriebssysteme

Den verschiedenen Hardwaresystemen entsprechend, haben wir es mit zwei unterschiedlichen Betriebssystemen zu tun. Das X-Oberon auf dem Targetsystem ist echtzeitfähig, das heisst, es stellt parallele Prozesse, Prioritäten und Mechanismen zur Prozesssynchronisierung sowie Interprozesskommunikation zur Verfügung. Die Vergabe der Prioritäten erfolgt für den Regelungstechniker in sehr intuitiver Weise (Bild 3). Das Betriebssystem kennt zeitkritische (TC) und nichtzeitkritische (NTC) Prozesse. Für zeitkritische Prozesse wird spezifiziert, wie häufig sie abgearbeitet werden sollen. Die Häufigkeit, mit der ein Prozess abgearbeitet werden soll, bestimmt zugleich dessen Priorität. Zum Beispiel kann ein Prozess, der nur alle 10 Sekunden abgearbeitet wird, regelmässig durch einen Prozess, der alle 10 Millisekunden abgearbeitet wird, unterbrochen werden. Die verbleibende Zeit steht für nichtzeitkritische Prozesse zur Verfügung. Für den Datenaustausch zwischen Prozessen können Variablen mit mehrseitigem Zugriff vor gleichzeitigem Zugriff geschützt werden. Für die Synchronisation von Prozessen stellt das Betriebssystem Signale zur Verfügung. Es war vor allem die Einfachheit des X-Oberon-Konzeptes, das es gegenüber dem in der Industrie weitverbreiteten Tornado/Vx-Works attraktiv machte.

Hostseitig dient Oberon System 3 als Betriebssystem [9]. Mit X-Oberon werden spezielle Schnittstellenbausteine (sogenannte Remote Gadgets) zur Verfügung gestellt, die sich mit entsprechenden Objekten auf dem Targetsystem verbinden lassen und so die Visualisierung von Vorgängen auf dem Target ermöglichen (auf die Verwendung der Remote Gadgets wird im Abschnitt «Komponieren von Benutzerschnittstellen» noch eingegangen).

Programmiersprache

Sowohl Target- als auch Hostcomputersystem sind mit Oberon programmierbar. Dabei können sowohl die Originalversion der Sprache [10] als auch der

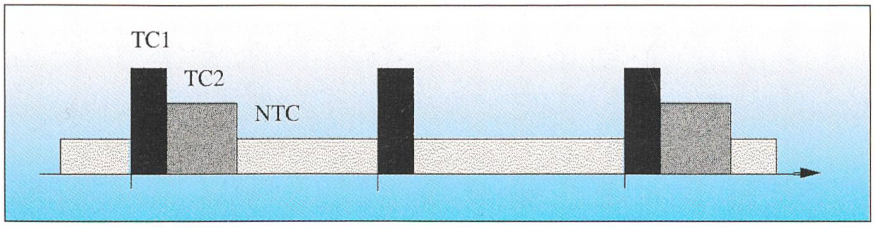


Bild 3 Beispiel für Prioritäten von Prozessen in X-Oberon
Die zeitkritischen Prozesse TC1 und TC2 haben unterschiedliche Periodizität, TC1 kommt häufiger zum Zug als TC2. Der Prozess TC1 hat demnach höhere Priorität als TC2. Die verbleibende, nicht durch zeitkritische Prozesse benötigte Rechenzeit steht für nichtzeitkritische Prozesse (NTC) zur Verfügung.

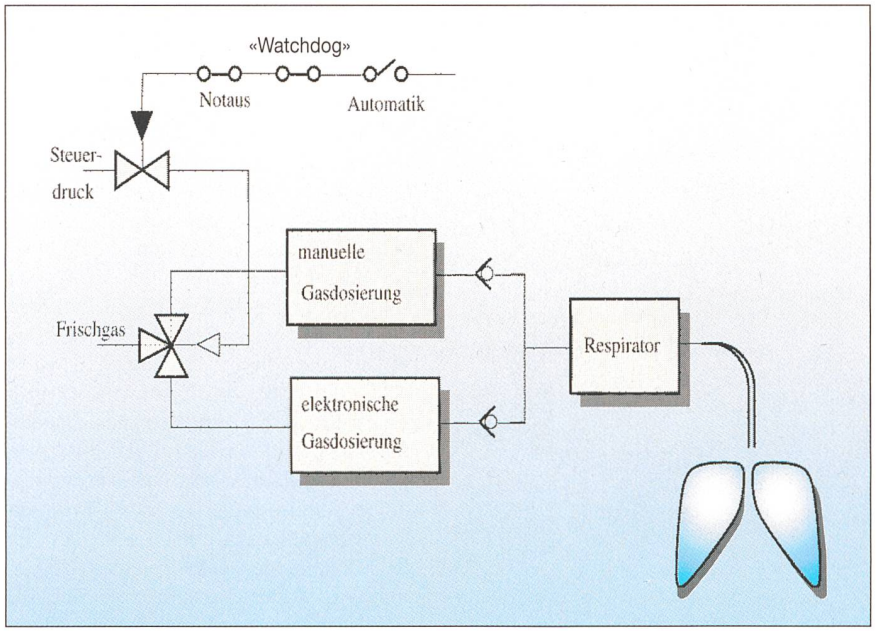


Bild 4 Fallback-Konzept

Dialekt Oberon-2 [11] verwendet werden. Die Sprache Oberon zeichnet sich durch ein klares, verglichen mit C++ einfaches Konzept aus, wie man es sich von Pascal und Modula gewohnt ist. Ein weiterer Vorteil sind die relativ strengen Checks zur Compilierzeit. Betrachtet man die Programmiersprache Java alleine, wäre auch sie eine Alternative, zum Zeitpunkt der Plattformwahl standen aber keine Java-programmierbaren Echtzeitsysteme zur Verfügung.

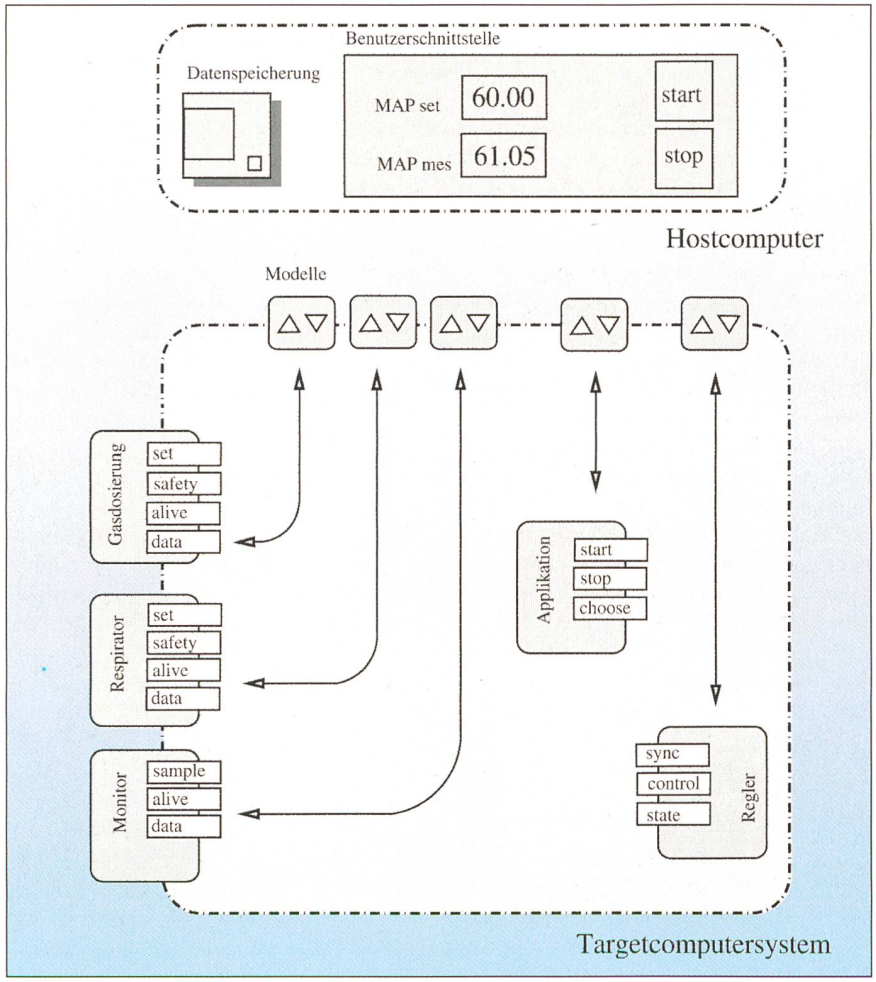


Bild 5 Softwarestruktur

Hardwaresicherheit

In der Fortsetzung des Projektes ist geplant, softwaregesteuerte Überwachungsfunktionen zu integrieren. Um bereits jetzt ein Maximum an Sicherheit zu gewährleisten, wird ein Konzept gemäss Bild 4 verwendet; es ist im Sinne einer Fallback-Strategie zu verstehen. Danach kann die Gasmenge entweder manuell oder elektronisch dosiert werden. Die Umschaltung erfolgt pneumatisch, und im Ruhezustand (drucklos und stromlos) ist immer die manuelle Dosierung aktiv. Die elektronische Dosierung ist nur dann aktiv, wenn Steuerdruck vorhanden ist, der Not-Aus-Knopf nicht gedrückt ist, der «Watchdog» nicht anspricht und die Regelung eingeschaltet ist. Sobald eine dieser Bedingungen nicht erfüllt ist, fällt dass System in manuellen Dosierbetrieb zurück, und zwar wird mit der letzten Dosiereinstellung gearbeitet, das heisst, dass der Anästhesist vor der Umschaltung auf automatische Regelung einen Preset-Flow einstellen kann, der aktiv wird, sobald etwas schiefgeht.

Softwarestruktur

Mit der geeigneten Wahl der Plattform ist ein wichtiger Schritt zur Erfüllung der oben aufgelisteten Anforderungen gemacht. In einem weiteren Schritt muss auch die Software entsprechend entworfen werden. Die Methoden für den Entwurf objektorientierter Software sind weit ausgereift [12, 13, 14] und unterscheiden sich nur unwesentlich. Alle Entwurfsmethoden verwenden Objektdiagramme, Klassendiagramme und eine Form von Interaktionsdiagrammen. Für die Echtzeitanwendung gilt es weiter, die einzelnen Tasks bzw. Prozesse zu definieren (vgl. [15]).

Am Anfang des Entwurfsprozesses steht immer eine Analysephase, bei der es darum geht, einen Überblick zu gewinnen. Nach [14] eignet sich dazu ein sogenanntes Context Picture, wie es in Bild 2 für das Anästhesiesystem wiedergegeben ist. Man ersieht daraus, dass im jetzigen Projektstadium mit drei medizinischen Geräten über je eine serielle Schnittstelle kommuniziert werden muss, wovon zwei dasselbe Protokoll benützen (Gasdosierung und Monitor). Zwei der Geräte sind Aktuatoren (Gasdosierung und Respirator) und eines ist ein Monitor. Host und Target kommunizieren via Ethernet-Verbindung. Weiter haben wir es mit drei

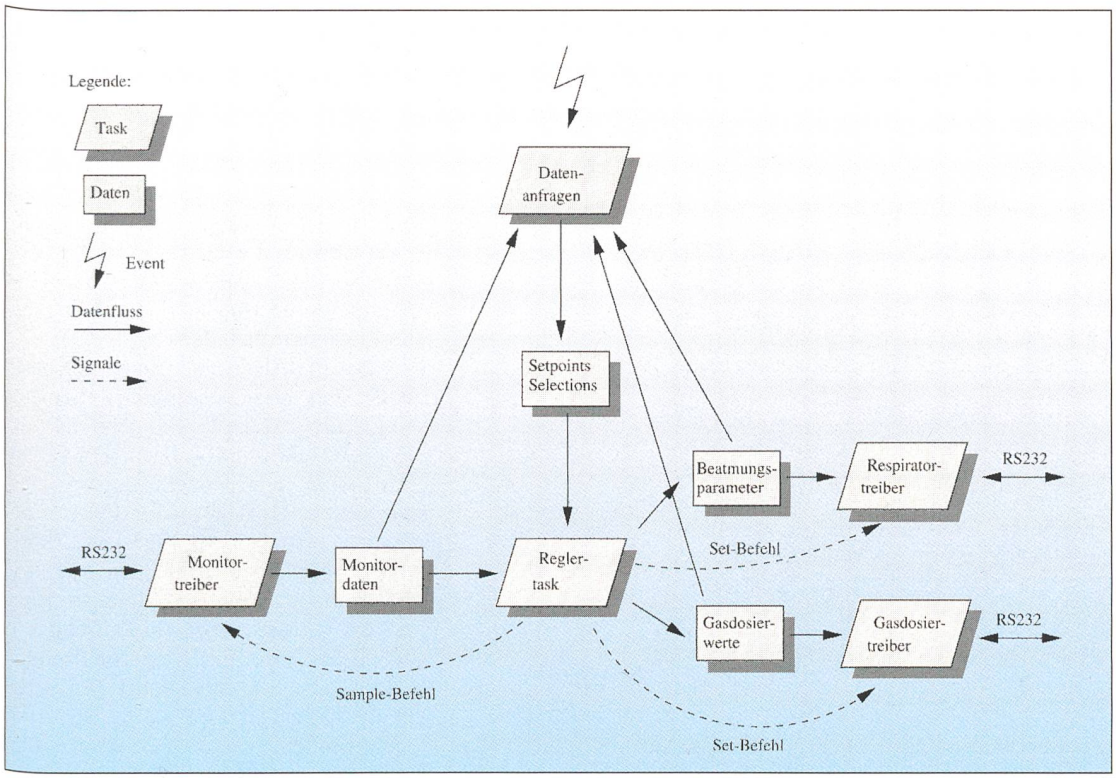


Bild 6 Das System kennt drei Klassen von Tasks: den Reglertask, Eingabe/Ausgabe-Tasks und Benutzer-schnittstellen-Interaktionen

Klassen von Benutzern zu tun, dem Anästhesisten, dem Systemingenieur und dem Programmierer, die unterschiedliche Anforderungen ans System haben. Schliesslich sollen die Daten auch abgespeichert werden.

Objektstruktur

Um zu einer sinnvollen Objektstruktur zu gelangen, kann man die folgenden Überlegungen anstellen. Aus der Sicht der Regler sind die entscheidenden Objekte die Ein- und Ausgangssignale, und im Prinzip spielt es für den Regler keine Rolle, ob der Wert von einem Gerät A oder von einem Gerät B gemessen wird. Ein möglicher Entwurf wäre demnach, Objekte der Klasse Messgrösse (für eine entsprechende Klasse Stellgrössen gelten analoge Betrachtungen) zu definieren, die den Reglern die aktuellsten Messwerte zur Verfügung stellen, sich die Werte von einem beliebigen Gerät, das dazu in der Lage ist, abholen können und in der Initialisierung überprüfen, ob ein entsprechendes Gerät vorhanden ist. Bei genauerer Betrachtung hinkt dieser Entwurf allerdings. Leider reicht es für den Entwurf eines Reglers nicht aus zu wissen, welche Messwerte oder Stellgrössen verfügbar sind, man braucht auch zusätzliche Informationen über die Messwerte und Stellgrössen. Bei Messwerten etwa die Genauigkeit, da jeder Regler irgendeine Filterung der Messwerte vornimmt und für den Entwurf eines guten Filters die Genauigkeit des Messwertes aus-

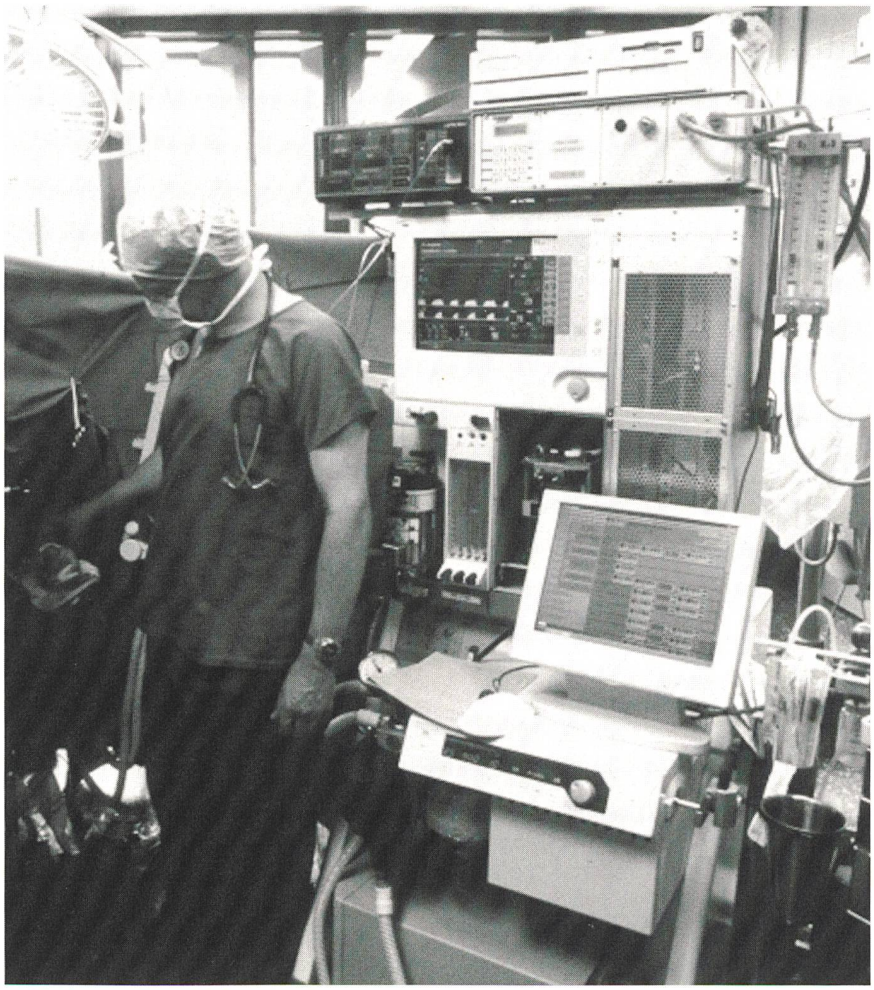


Bild 7 Das experimentelle System im Einsatz

Alle für die Implementierung von automatischen Reglern notwendigen Komponenten konnten auf einem gängigen Respirator untergebracht werden. Daher ist die Anlage kompakt und wie gewohnt zu bedienen.

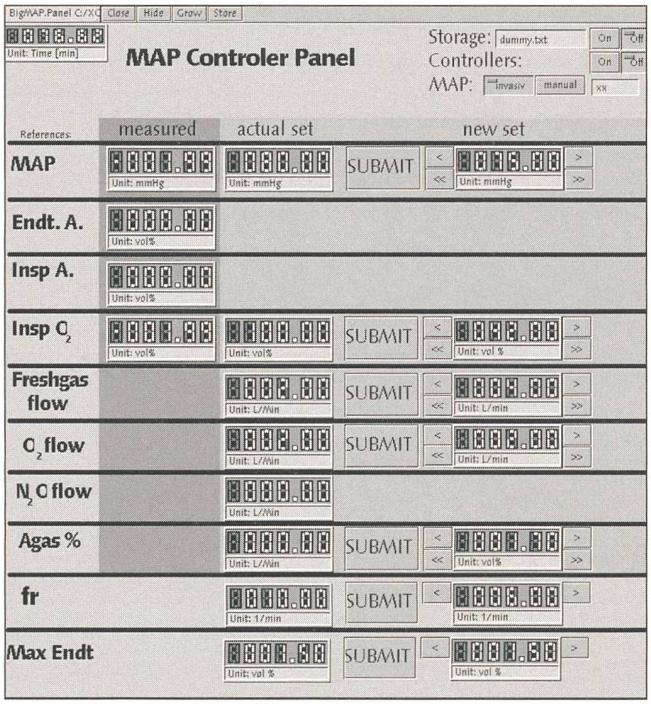


Bild 8 Beispiel einer Prototyp-Benutzerschnittstelle für einen kombinierten Blutdruck/Sauerstoff-Regler

schlaggebend ist. Im weitern ist die Vertrauenswürdigkeit einer Messung oder auch die Reaktionszeit des Messgerätes auf eine Änderung der Messgrösse beim Reglerentwurf von Bedeutung, ebenso in welchen Intervallen ein Messwert verfügbar ist. Das gleiche Problem zeigt sich auch bei Stellgliedern. Wenn ein Stellglied einen Stellbefehl nicht sofort ausführen kann, müssen diese Verzögerungszeiten beim Reglerentwurf mitberücksichtigt werden, ebenso Beschränkungen des Stellbereichs und andere Eigenschaften. Diese Überlegungen führen dazu, dass es am sinnvollsten ist, die Wirklichkeit 1:1 in Objekte abzubilden. Bild 5 zeigt die definierten Objekte auf höchster Ebene. Wir haben also für jedes angeschlossene Gerät und für jeden Regler ein Objekt. Wir haben weiter ein zentrales Objekt, das wir Applikation nennen und dessen Aufgabe noch beschrieben wird. Jedes Objekt verfügt über ein Modell, welches es erlaubt, grafische Elemente der Benutzerschnittstelle auf dem Hostcomputer mit Signalen zu verbinden. Dieselben Modelle werden auch vom Objekt verwendet, das für die Abspeicherung zuständig ist. Wir werden hier nicht weiter auf Objekt- oder Klassenstruktur eingehen, sondern uns den einzelnen Tasks widmen.

Taskstruktur

Bild 6 zeigt die Taskstruktur des Systems, die Notation ist [15] entnommen. Es lassen sich drei Typen von Tasks erkennen. Der zentrale Task ist der Reglertask, der über das Applikationsobjekt definiert ist und der für das Einlesen vom Monitor, für das Aufrufen der Reglerobjekte und das Auslösen der Ausgabe verantwortlich ist. Dieser Task ist als zeitkritischer Prozess definiert, der alle 10 Sekunden abläuft. Mit anderen Worten, der Reglertakt ist etwas langsamer als typische Atemzyklen. Als weitere Tasks treten die Gerätetreiber auf. Es ist nötig, diese Treiber als separate Tasks zu definieren, da die Geräte unterschiedlich häufig angesprochen werden müssen, weil sie sonst aus dem Fernbetriebsmodus in den Handbetrieb zurückfallen oder Alarme abgeben. Die Gerätetreiber sind ebenfalls als periodische Prozesse definiert. Die Geräteschnittstellenobjekte und das Applikationsobjekt in Bild 5 können demnach als aktive Objekte aufgefasst werden. Die letzte Klasse von Tasks sind die Datenanfragen der Schnittstellenelemente des Hosts. Diese Anfragen werden durch Refresh- oder Touchscreen-Befehle asynchron ausgelöst und werden auf dem Target als nichtzeitkritische Prozesse behandelt.

Schreiben einer Applikation

Die Eingabe/Ausgabe-Objekte und deren Modelle sind auf der Plattform verfügbar. Applikationsobjekt und dessen Modell sind als Gerüst vorhanden. Die Plattform stellt somit ein Framework zur Implementierung von automatischen Reglern zur Verfügung, bei dem der Anwender (Regelungstechniker) bloss einige wenige Zeilen Code in das Applikationsobjekt einfügen muss, die den Ablauf bestimmen. Es ist daher sehr einfach, den periodischen Reglertask zu ersetzen durch einen Reglertask, der auf externe Abläufe (wie z.B. Beatmungszyklus) synchronisiert ist. Daneben müssen auch die Reglerobjekte sowie deren Modelle entworfen werden. Beschleunigt wird die Implementierungsphase durch das dynamische Laden und Linken der Oberon-Module. Auf diese Art kann auch ohne integrierten Debugger in X-Oberon effizient entwickelt werden.

Komponieren von Benutzerschnittstellen

Zum Erstellen einer Benutzerschnittstelle sind spezielle grafische Elemente (sogenannte Remote Gadgets) auf der Hostseite und die Modelle auf der Targetseite nötig. Die Modelle dienen dabei einzig dazu, das typische Format der Oberon System 3 Messages in entsprechende Methodenaufrufe der Targetobjekte umzuwandeln. Im Moment stehen auf der Hostseite etwa ein Dutzend verschiedener Anzeigeelemente wie Knöpfe, Sliders, Siebensegmentanzeigen, 2D-Plots usw. zur Verfügung, um Benutzerschnittstellen zu gestalten. Diese Elemente können aus einer Bibliothek geladen, auf dem Bildschirm positioniert und mit Variablen der targetseitigen Modelle verbunden werden. Ein Beispiel für eine Benutzerschnittstelle gibt Bild 8 wieder. Nach dem Komponieren kann das Panel in den Zustand «locked» gebracht werden, so dass keine versehentlichen Änderungen am System möglich sind und der bedienende Anästhesist lediglich die für die Reglerbedienung vorgesehenen Befehle ausführen kann.

Zusammenfassung

In diesem Aufsatz wurde der Entwurf eines Gesamtsystems beschrieben, das es erlaubt, automatische Regler für die Anästhesie im Operationssaal zu testen. Es wurde dargelegt, welche Überlegungen zur aktuellen Plattformwahl geführt haben und wo die Vorteile der einzelnen Komponenten liegen. Bei der Softwarestruktur wurde spezielles Gewicht auf die Erweiterbarkeit des Systems und die Auswechselbarkeit von Softwareteilen gelegt. Der Aufbau erlaubt es nun, ohne besondere Programmierkenntnisse Applikationen für das System zu realisieren und die Vorgänge zu visualisieren.

Verdankung

Die Autoren danken dem Schweizerischen Nationalfonds, der Firma Dräger und der Stiftung für die Förderung der wissenschaftlichen Forschung der Universität Bern für die grosszügige Unterstützung.

Literatur

[1] *A. M. Zbinden, P. Feigenwinter, S. Petersen-Felix and S. Hacisalihzade:* Arterial pressure control with isoflurane using fuzzy logic. British Journal of Anaesthesia 74(1995), pp. 66–72.

[2] *M. Curatolo, M. Derighetti, S. Petersen-Felix, P. Feigenwinter, M. Fischer and A.M. Zbinden:* Fuzzy logic control of inspired isoflurane and oxygen concentrations using minimal flow anaesthesia. British Journal of Anaesthesia 76(1996), pp. 245–250.

[3] *J. Schäublin, M. Derighetti, P. Feigenwinter, S. Petersen-Felix and A. M. Zbinden:* Fuzzy logic control of mechanical ventilation during anesthesia. British Journal of Anaesthesia 77(1996)5, pp. 636–641.

[4] *D. G. Mason, D. A. Linkens, N. D. Edwards and C. S. Reilly:* Development of a portable closed-loop atracurium infusion system: system methodology and safety issues. International Journal of Clinical Monitoring, 13(1997), pp. 243–252.

[5] VDI (Verein Deutscher Ingenieure): Konzepte fehlertolerierender Automatisierungssysteme. VDI/VDE-Richtlinie 3698, Juli 1995.

[6] *N. Leitgeb:* Sicherheit in der Medizintechnik. Renningen-Malmsheim, Expert-Verlag, 1995.

[7] *W. A. Halang und R. Konakovsky:* Sicherheitsgerichtete Software. Automatisierungstechnik 46(1998)2, S. 93–103.

[8] *C. W. Frei und D. Leibundgut:* Wahl einer Plattform für ein automatisiertes Anästhesie-System. Technical Report Nr. AUT 97-03, Automatic Control Lab, Swiss Federal Institut of Technology (ETH), Februar 1997.

[9] *A. Fischer and H. Marais:* The Oberon Companion. Zürich, Vdf-Hochschulverlag, 1998.

[10] *M. Reiser and N. Wirth:* Programming Oberon, Steps beyond Pascal and Modula. Reading: Addison-Wesley, 1992.

[11] *H. Mössenböck:* Object-Oriented Programming in Oberon-2. Berlin/Heidelberg, Springer-Verlag, 1995.

[12] *J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy and W. Lorensen:* Object-Oriented Modeling and Design. Prentice Hall International Editions, 1991.

[13] *Grady Booch:* Object-Oriented Analysis and Design with Applications. The Benjamin/Cummings Publishing Co., 1994.

[14] *S. Sigfried:* Understanding Object-Oriented Software Engineering. IEEE Press, 1996.

[15] *H. Gomaa:* Software design methods for concurrent and real-time systems. Reading: Addison-Wesley, 1993.

Une plate-forme pour systèmes automatisés en anesthésie

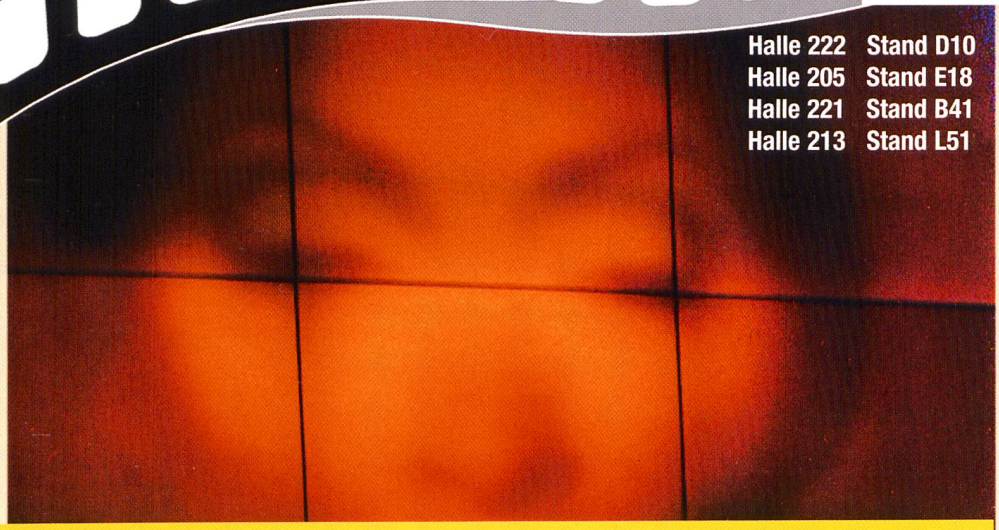
L'exécution d'une narcose pose à l'anesthésiste des problèmes notamment au niveau du réglage: pendant une intervention chirurgicale il doit enregistrer les données des fonctions corporelles vitales et les apprécier en les comparant à des valeurs de consigne appropriées. Le médecin doit compenser les écarts de consigne par un réglage manuel. A l'hôpital de l'Ile, Berne, on développe conjointement avec l'EPF Zurich un système qui est censé décharger l'anesthésiste d'une partie de ses tâches, en lui retirant celles de réglage proprement dites pour qu'il se consacre plus à la surveillance.

L'article décrit le projet d'un tel système total. Le système permet de tester le régulateur automatique pour l'anesthésie en salle d'opération. Au niveau de la structure du logiciel, on a attaché une importance particulière à l'évolutivité du système et à l'interchangeabilité de parties du logiciel. La structure permet de réaliser des applications pour le système sans pour autant devoir disposer de connaissances en programmation.

Einladung zur Film-Premiere

1.–4.9.1998
S.A.W.
SWISS AUTOMATION WEEK
Messe Basel.

Halle 222 Stand D10
Halle 205 Stand E18
Halle 221 Stand B41
Halle 213 Stand L51



Ihr Cinema
Cinema Gratis-Ticket zum
«Rockwell Automation Cinema»
Cinema

Ja! **«Automation World» «Motor Management»** – sowie weitere ausser-gewöhnliche Präsentationen feiern Premiere im Rockwell Automation Cinema an der S.A.W. in Basel. Film, Sound, Laser und Life-Präsentationen reissen Sie für kurze Zeit aus dem Messealltag und entführen Sie in die phantastische Welt der industriellen Automation. Rockwell Automation und die Mitaussteller AarSys, bürge-fischer, EME, Induma, Inspectron, Knellwolf + Frehner, Schaltag heissen Sie zur **S.A.W.** in **Halle 222, Stand D10** herzlich willkommen und verheissen Ihnen schon jetzt Hochspannung und Action.

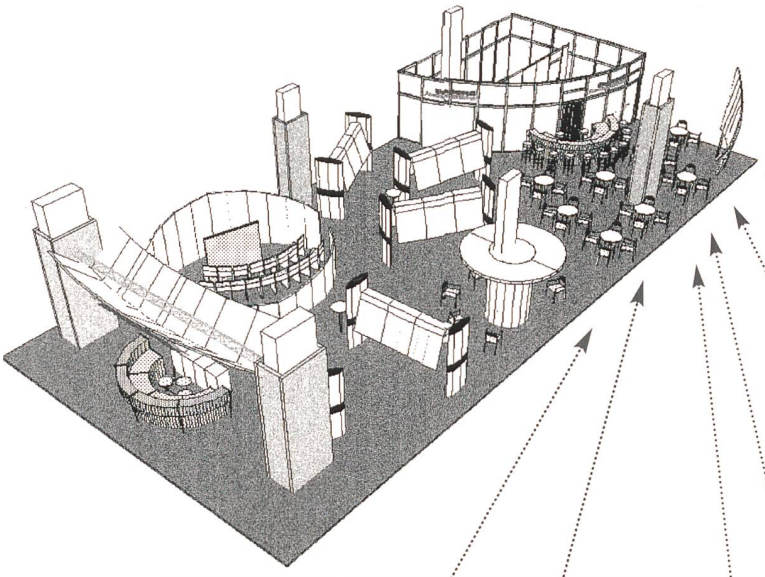
k+f
bürge-fischer
AarSys AG
AUTOMATION & UMWELTTECHNIK
Inspectron
Induma
seit 1977
SCHALTAG
EME

Rockwell Automation AG
Verkaufs- und Supportzentrum
Gewerbepark, 5506 Mägenwil
Telefon 062 889 77 77
Telefax 062 889 77 66

Allen-Bradley
RELIANCE ELECTRIC
DODGE
ROCKWELL SOFTWARE

Rockwell Automation

Rockwell Automation und die Mitaussteller heissen Sie herzlich willkommen.



Rockwell Automation AG
Verkaufs- und Supportzentrum
Gewerbepark, Postfach 64
5506 Mägenwil
Tel. 062/ 889 77 77
Fax 062/ 889 77 66

Induma AG
Bächigenstrasse 18, 9212 Arnegg
Tel. 071/385 64 66
Fax 071/385 64 68
Oberemattstrasse 18, 4133 Pratteln
Tel. 061/821 80 00
Fax 061/821 80 07
Ingenieurbüro mit Network für Automation, Leittechnik, Netzwerke und EDV

AarSys AG
Vogelsang 73, 2502 Biel
Tel. 032/322 64 62
Fax 032/322 64 34
Automation und Umwelttechnik

Inspectron AG
Böhnirainstrasse 13
8800 Thalwil
Tel. 01/722 72 72
Fax 01/722 72 22
Qualitätssicherungssysteme in der industriellen Bildverarbeitung

bürge-fischer ag
Mattenstrasse 3, 5745 Safenwil
Tel. 062/789 85 85
Fax 062/789 85 86
Ein starker Partner für die Industrieautomation

Knellwolf + Frehner AG
Schwellbrunnerstrasse 85
9100 Herisau
Tel. 071/353 05 05
Fax 071/353 05 95
Elektrische und pneumatische Steuer- und Regelanlagen/Schaltanlagenbau

EME AG
Interconnection & Motion
Lohwisstrasse 50, 8123 Ebmatingen
Tel. 01/982 11 11
Fax 01/982 11 22
Lokales Kompetenzzentrum für Motion Control

Schaltag AG
Industriestrasse 8, 8307 Effretikon
Tel. 052/354 27 27
Fax 052/354 27 21
Systemanbieter für Steuerungs- und Antriebstechnik

Besuchen Sie uns und unsere Partner auch an diesen weiteren Standorten.

Standorte Rockwell Automation:			Standorte Partnerfirmen von Rockwell Automation:		
Hauptstand Rockwell Automation	Halle 222	Stand D10	bfa Büro für Automatisierung	Halle 221	Stand D40
Demovan	Halle 205	Stand E18	EHS Schaffhausen AG	Halle 222	Stand D15
Forum Software & Services	Halle 221	Stand B41	Schaltag AG	Halle 222	Stand B22
Gemeinschaftsstand Sensorik	Halle 213	Stand L51	Swibox	Halle 222	Stand B20