Remote-Administration von eingebetteten Systemen mit einem Java-basierten Add-On-Modell

Frank Burchert, Christian Hochberger, Ulrike Kleinau, Djamshid Tavangarian

Universität Rostock, Institut für Technische Informatik
Lehrstuhl für Rechnerarchitektur
Albert-Einstein-Straße 21, 18059 Rostock
Tel. +49 (0)381 / 498 - 3386 Fax: -3440
E-Mail: ra@informatik.uni-rostock.de

Kurzfassung. Dieser Beitrag behandelt prinzipielle Architekturen für die Fernadministration und -wartung eingebetteter Systeme und stellt eine neue, modulare Lösung auf der Basis eines Add-On-Modells vor, mit der sich insbesondere bereits bestehende Systeme zur sicheren Administration über Intranet oder Internet erweitern lassen. Der Einsatz von Java zeigt sich dabei als probates Mittel, um die Sicherheit zu erhöhen und diese Systeme auf einfache Weise via Web fernzubedienen. Authentifizierung, Autorisierung und Sicherstellung der Vertraulichkeit sind dabei die Schlüssel zur Realisierung solcher Systeme.

1 Einleitung

Wie die Vergangenheit gezeigt hat, profitieren eingebettete Systeme stets von den Entwicklungen auf den Gebieten der Soft- und Hardware für Desktop-Systeme bzw. Workstations, wenn auch mit einer gewissen zeitlichen Verzögerung. Das gilt auch für die Anbindung eingebetteter Systeme an ein Kommunikationsnetz: Während bei Arbeitsplatzrechnern die Anbindung der Systeme an Netzwerke schon längst ein fester Bestandteil heutiger EDV-Konzeptionen geworden ist, hat sie bei den eingebetteten Systemen gerade erst begonnen. Die Vorteile einer solchen Anbindung liegen dabei klar auf der Hand: Neben zahlreichen Möglichkeiten für die Applikationen selbst (man denke hier beispielsweise an die Programmierung verteilter Systeme) sind das insbesondere Dienste wie Fernadministration, Fernwartung und -überwachung, die hinzukommen. Dabei werden nicht nur Kosten gesenkt, wenn für die Wartung oder Administration nicht mehr ein Techniker vor Ort sein muß, sondern es ist auch ein Zugewinn an Service, wenn diese Dienste von jedem Ort aus und damit von wenigen Experten für den Anwender vorgenommen werden können, ohne daß dieser sich mit der Administration selbst auskennen muß [6]. Da ein Großteil der heutigen Funktionalität eingebetteter Systeme in Software realisiert ist, bietet sich hier ferner die Möglichkeit, auf einfache Art Updates vorzunehmen, mit denen nachträglich eine neue oder geänderte Funktionalität implementiert werden kann.

Da ein über Internet erreichbares System aber auch verletzlicher gegenüber Angriffen aus dem Netz ist, sind eine Reihe von Maßnahmen zu treffen, die den sicheren Betrieb gewährleisten sollen und bereits bei der Architektur solcher Systeme zu berücksichtigen sind. Dieser Beitrag beschäftigt sich daher mit drei zentralen Fragen, die sich im Zusammenhang mit der Anbindung eingebetteter Systeme an das Internet immer wieder stellen:

- Welche Aufgaben bestehen im Zusammenhang mit der Administration bzw. dem Management[1] eines Systems im laufenden Betrieb und welche können davon aus der Ferne übernommen werden?
- Welche architektonischen Maßnahmen sind zur Lösung dieser Aufgaben erforderlich?
- Welche Sicherheitsaspekte sind bei der Fernadministration prinzipiell zu berücksichtigen?

Der Aufbau dieses Beitrags gliedert sich hierzu folgendermaßen: Während Abschnitt 2 kurz den gegenwärtigen Stand bei der Remote-Administration von (eingebetteten) Systemen vorstellt, geht der darauffolgende Abschnitt 3 auf die allgemeinen Anforderungen und die damit im Zusammenhang stehenden Architekturentscheidungen ein. Abschnitt 4 stellt eine daraus abgeleitete neuartige Lösung für Systeme heraus, die mit einem Add-On-Modul zur Remote-Administration befähigt werden. Ein Add-On-Modul stellt dabei eine speziell für diese Aufgabe qualifizierte, eigenständige Hard- und Software-Einheit auf der Basis von Java dar. Die Abschnitte 5 und 6 zeigen auf, welche Vorteile sich hierbei durch die Verwendung der Programmiersprache Java ergeben und wie die Sicherheit vor unzulässigen Eingriffen in das eingebettete System durch javabasierte und andere Maßnahmen gewahrt werden kann.

2 Remote-Administration und -Management: Stand der Technik

2.1 Fernsteuerung und -überwachung eingebetteter Systeme

Ferngesteuerte, -überwachte oder auch -administrierbare Systeme gibt es bereits seit längerer Zeit. Beispiele hierfür finden sich sowohl im industriellen als auch im privaten Bereich. Exemplarisch seien hier Telekommunikationsanlagen, Anrufbeantworter oder auch die moderne Haustechnik genannt, wo verschiedenste Hausgeräte, die beispielsweise an einem seriellen Bus angeschlossen sind, ferngesteuert bzw. -überwacht werden können. In der Regel werden dabei zur Kommunikation herkömmliche Telefonverbindungen genutzt, über die sich der Anwender in das System einwählt, das dann durch eine spezielle Software administriert oder auch einfach nur überwacht werden kann. Meist wird bei der Verbindung zum eingebetteten System ein proprietäres Kommunikationsprotokoll genutzt. Nachteilig an dieser Art der Fernadministration ist, daß der unberechtigte Zugang hier in der Regel lediglich durch ein Paßwort und das ggf. proprietäre Protokoll verhindert wird. Die Kommunikation selbst wird in der Regel unverschlüsselt durchgeführt. Steht das zu administrierende Gerät weit entfernt, fallen darüber hinaus unter Umständen hohe Verbindungsgebühren an.

IP-basierte Remote-Administration und -steuerung von eingebetteten Systemen steht zur Zeit noch am Anfang der Entwicklung (Bsp.: steuerbare Web-Kameras, ...). Sie wird erschwert durch die große Anzahl von Plattformen und Architekturen, die in diesem Bereich zu finden sind. Hierzu einen weitestgehenden generischen und zukunftsträchtigen Ansatz zu finden, ist eine aktuelle Aufgabe, für die es bereits erste Ansätze gibt [4][6]. Die Bedeutung des Einsatzes der plattformunabhängigen Web-Technologien (von HTML bzw. XML über HTTP bis Java) wird wegen der damit verbundenen Vorteile (s. Abschnitt 5) mittlerweile nicht mehr bestritten. In allen Ansätzen spielen Applets bzw. Servlets eine zentrale Rolle; die Benutzeroberfläche wird in der Regel wegen der Flexibilität von Web-Browsern gebildet.

[1] In diesem Beitrag werden die Begriffe "Administration" und "Management" synonym gebraucht.

2.2 Standardisierte Verfahren zur Administration netzgebundener Systeme

Die Fernadministration von Netzwerken bzw. deren Komponenten ist im Gegensatz zur Situation der eingangs erwähnten eingebetteten Systeme schon sehr viel fortschrittlicher. Hierzu wurden in der Vergangenheit verschiedene Management-Werkzeuge geschaffen, die dem Administrator die Möglichkeit geben, eine große Anzahl von Maschinen und Peripheriegeräten über Local Area Networks (LANs) und Wide Area Networks (WANs) zu verwalten. Ein klassisches Beispiel ist hier das standardisierte, herstellerunabhängige Simple Network Management Protocol (SNMP), das in seinen verschiedenen Versionen SNMPv1 bis SNMPv3 bereits eine ansehnliche Verbreitung gefunden hat. Das Management fußt hier auf sogenannten Management Information Bases (MIBs), die ein jedes zu administrierende Gerät über eine Anzahl von spezifizierenden Parametern beschreiben und durch einfache GET und SET-Aufrufe der aufsetzenden SNMP-Management-Software abgefragt bzw. modifiziert werden können. Über Traps können (Alarm-) Meldungen vom administrierten System an die Management-Station gereicht werden. Der SNMP-Agent übernimmt dabei die Vermittlerrolle zwischen dem Management-System auf der einen und der zu managenden Ressource auf der anderen Seite und gehört damit zusammen mit dem Management-System zu den handelnden Instanzen in diesem Modell (s. Abb. 1) [5].

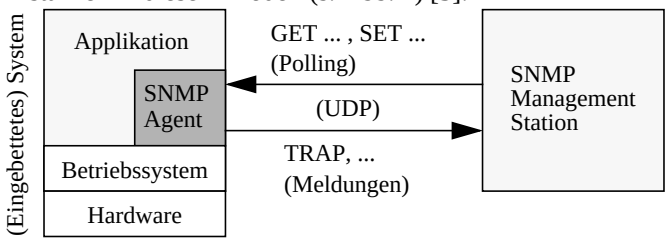

Abb. 1. SNMP-basiertes Remote-Management

Authentifikation und Autorisierung gehören ebenso wie die Verschlüsselung mittlerweile zu einem unverzichtbaren Standard, wenn sichere Verbindungen über IP-Netze realisiert werden sollen. In SNMPv2 wurden diese Sicherheitsmerkmale daher weitestgehend übernommen. Der gravierende Nachteil, daß SNMP-Daten nach wie vor über das unsichere UDP (User Datagramm Protocol) verschickt werden und damit während des Transports verlorengehen können, besteht aber immer noch.

2.3 Neue Entwicklungen im (Remote-) Management-Umfeld

Ein möglicherweise neuer Standard auf dem Gebiet der netzbasierten Administration verspricht die von verschiedenen Herstellern unterstützte Initiative „Wired for Management“ (WfM) zu werden, mit dem selbst Systeme ohne ein installiertes Betriebssystem durch eine pre-boot-Verbindung zu einem im BIOS oder Flash-ROM befindlichen Service-Agenten via Netzverbindung installiert und konfiguriert werden können. Für WfM sind derzeit sowohl On-Board- als auch PCI-Steckkartenlösungen innerhalb der PC-Welt vorgesehen. Damit kann ein System auch nach einem zerstörerischen Totalabsturz über Netz neu installiert werden, sofern eine Verbindung zu ihm aufgebaut werden kann.

Die Jini-Technologie macht es möglich, daß sich Geräte spontan in ein Netzwerk einklinken, um dort unmittelbar anderen Netzteilnehmern ihre Dienste anzubieten. Damit wird eine hochdynamische und -skalierbare Kommunikationsinfrastruktur angestrebt. Die grundsätzliche Fragestellung, wie sich diese Geräte sicher aus der Ferne

administrieren lassen, bleibt dabei aber bestehen: Auch derart spontan eingebundene Geräte erlauben in der Regel individuelle Konfigurationen, die für die jeweilig gewünschte Funktionalität explizit einzurichten sind. Die Firma Sun bietet in diesem Zusammenhang ein Java Dynamic Management Kit (JDMK) an, das in der Form von JavaBeans-Komponenten Grundelemente zum Aufbau von Managementsystemen bietet, die auf der Basis von dynamischen, servicefordernden Managementagenten und webbasierten, servicepropagierenden Managementinstanzen agieren [8]. Dabei wird neben RMI (Remote Method Invokation) u.a. auch das SNMP-Protokoll unterstützt. Sicherheitsaspekte scheinen aber noch ein kritischer Punkt zu sein.

Mit der zunehmenden Verbreitung und Verfügbarkeit des Internets an jedem beliebigen Ort wird die Fernadministration über IP-Netze auch für eingebettete Systeme interessant. Der folgende Abschnitt führt an, welche Anforderungen dabei an solche Systeme zu stellen und welche Architekturen bereits mit den heutigen Techniken realisierbar sind. Dabei wird aufgezeigt, wo der Einsatz von Java sinnvoll erscheint.

3 Architekturen fernadministrierbarer und -wartbarer Systeme

Wenn eingebettete Systeme über IP-Netze fernadministrierbar sein sollen, müssen sie zusätzlichen Anforderungen genügen, die es bereits bei der Architektur zu berücksichtigen gilt. Dazu gehören vor allem spezielle Sicherheitsmaßnahmen für das Zugangssystem, die mit Mitteln wie Authentifizierung, Autorisierung und Verschlüsselung oder Nutzung digitaler Signaturen zu erfüllen sind (s. Abschnitt 6). Zu den Aufgaben, die damit prinzipiell aus der Ferne übernommen werden können, zählen

- Administration, Konfiguration und Wartung des Systems, wozu auch die Aktualisierung (Updating) bzw. der Austausch von Software-Komponenten (Treiber, Applikationen, Systemsoftware, ...) gezählt wird ,
- Modifikation von Systemparametern,
- Überwachung (Monitoring) von Betriebsdaten des eingebetteten Systems und seiner Komponenten (wie z.B. Sensoren, Aktoren),
- Implementation von Test- oder Service-Software zur Fehlererkennung im Störungsfall.

Durch die Anbindung eingebetteter Systeme an ein Kommunikationsnetz gewinnen diese die Möglichkeit,

- autonom Betriebsdaten oder Störungen zu melden sowie
- Wartungsanforderungen in Abhängigkeit von der realen Benutzung des Systems abzusetzen.

Fernadministrierbare Systeme weisen grundsätzlich eine Architektur auf, die sich grob in eine Zugangs- und eine Management-Komponente teilt, die im einzelnen sehr unterschiedlich ausgestaltet sein können. In der letzen Zeit kristallisieren sich für das Zugangssystem dabei immer häufiger Web-Techniken (HTML, XML, SHTTP, Java, ...) heraus, was aber nicht als Beschränkung aufgefaßt werden sollte. Applets spielen hier eine zunehmende Rolle, da mit ihnen der Aufwand auf Client-Seite im wesentlichen auf das Vorhandensein eines Web-Browsers reduziert werden kann. Auf der Seite des Managementsystems (Client-Seite) werden gelegentlich Datenbankdienste verlangt, wenn eingebettete Systeme z.B. in unterschiedlichen Konstruktionsvarianten vorliegen, die dort abgelegt sind. Java bietet mit seinen zahlreichen APIs gute Möglichkeiten der plattformunabhängigen Anbindung solcher Systeme (z.B. mittels JDBC).

Fernadministrierbare eingebettete Systeme lassen sich in zwei wesentliche Kategorien einteilen, die hier als *integrierte* und *modulare*[2] Systeme bezeichnet werden sollen. Während bei den ersteren keine organisatorische Trennung der Remote-Administration von den Komponenten des eigentlichen eingebetteten Systems festzustellen ist (Gesamtsystem als monolithischer Block), ist die Fernadministration in letzteren durch organisatorisch separate Module realisiert. Das hat den Vorteil, daß zukünftige Innovationen der Techniken für das Zugangssystem berücksichtigt werden können, indem lediglich die betreffende Komponente getauscht wird. Eingriffsmöglichkeiten in das eingebettete System bestehen dabei über standardisierte Schnittstellen.

Für integrierte Systeme existiert bereits ein weitestgehend generischer Ansatz, mit dem das Management eingebetteter Systeme in das allgemeine Netzwerkmanagement eingebunden werden kann [7]. Darin wird eine Backplane-Architektur vorgeschlagen, die in der Form einer zusätzlichen Adaptionsschicht einen universellen Zugang zu dem eingebetteten System und seinem Innenleben eröffnet. Dieses Zugangssystem realisiert dabei auch die wichtigen Sicherheitsmaßnahmen (s. Abb. 2a). Nachteilig an dieser Architektur ist, daß es nach wie vor eine enge Verknüpfung zwischen dem eigentlichen eingebetteten System und dem Zusatz zur Fernadministration gibt, der sich in der Notwendigkeit des Gluecodes widerspiegelt.

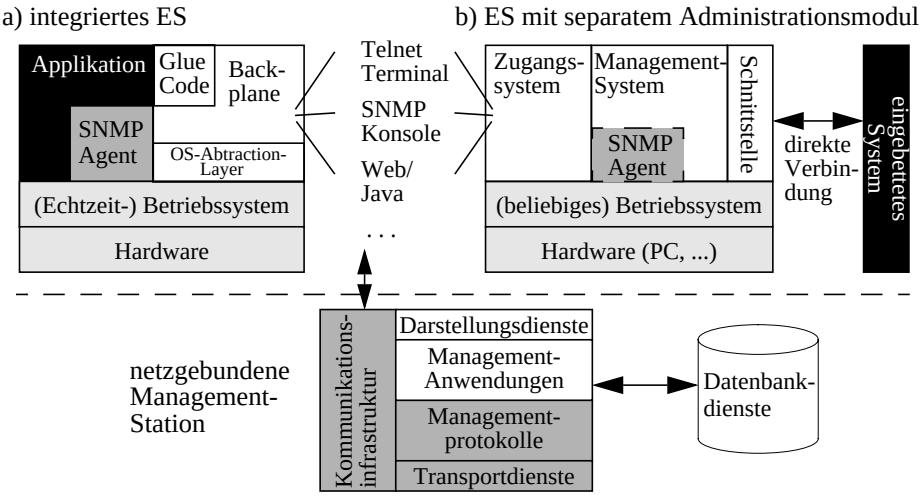

Abb. 2. Die modular konzipierte Fernadministration für eingebettete Systeme (ES)

Ein etwas anderer Ansatz wird bei den Add-On-Administrationsmodulen verfolgt, mit denen auch bereits existierende eingebettete Systeme zur IP-basierten Fernadministration erweitert werden können. Derartige Module verfügen über eigene Hard- und Software-Komponenten zur Erfüllung der Aufgabe und sind über eine direkte Verbindung mit dem eigentlichen eingebetteten System gekoppelt. Diese Verbindung wird in der Regel über eine Standardschnittstelle erfolgen. Die Zugangsschicht sorgt auch hier für die Realisierung der erforderlichen Sicherheitsmaßnahmen. Sie läßt sich gut mit Java-Applets realisieren, die aufgrund ihrer Eigenschaften für eine zusätzliche Zugangssicherheit sorgen (vgl. Abschnitte 4 und 6). Über die dargestellte Architektur wird es mit Add-On-Modulen darüber hinaus ermöglicht, bislang nicht SNMP-fähige Syste-

[2] Der Begriff "modular" soll hier nicht im Sinne des Software-Engineering verstanden werden, sondern im Sinne separater, aus Hard- und Software bestehender Module.

me in das vorhandene Netzwerkmanagement einzubinden, wobei dieses Merkmal nicht auf SNMP-basiertes Management beschränkt sein muß (s. Abb. 2b).

Nicht aufgeführt sind in der Abbildung die architektonischen Maßnahmen, die zur Selbstüberwachung ergriffen werden müssen. Dies können in der einfachsten Realisierung Watchdog-Timer sein, die für einen Neustart sorgen, wenn das System längere Zeit in einem Prozeß hängen bleibt.

Zu den kritischen Aufgaben gehört die Fernadministration des Basissystems, das den Systemzugang über Netz sicherstellt, da hier grundsätzlich die Gefahr besteht, „an dem Ast zu sägen, auf dem man gerade sitzt“. Eine Lösung hierzu bietet die Anwendung eines *Fall-Back-Verfahrens*, das dafür sorgt, daß eine fehlgeschlagene Neukonfiguration des Systems nach einer gewissen Zeit zur automatischen Reaktivierung der alten Konfiguration führt, die zu diesem Zweck gespeichert bleiben muß. Im anderen Fall kann der Administrator diese Reaktivierung per Befehl unterbinden, nachdem das System erfolgreich eine neue Verbindung zu ihm aufbauen konnte.

Wenn ausschließlich ein temporärer Zugang zum Management-Server des eingebetteten Systems benötigt wird, dann kann dessen Verbindung zum Internet auch über einen zwischengeschalteten Proxy-Server geschehen, damit das zu administrierende Gerät nicht eine dauerhafte Verbindung zum Internet aufrechterhalten muß, die je nach Verbindungsart u.U. mit erheblichen Kosten verbunden sein kann. Bei der Anwahl eines Systems wird dann vom Proxy aus eine Verbindung zum eigentlichen Server aufgebaut, um die Daten durchreichen zu können.

4 Separates Administrationsmodul: Architektur und Komponenten

Die Administrationsfunktionalität kann in einem separaten Modul realisiert werden (s. Abb. 3). Die Administration erfolgt hierbei über das Internet unter Einsatz eines Browsers als Client-Software. Der Browser lädt vom Web-Server des Moduls eine HTML-Seite, die wiederum ein spezielles Applet referenziert, das dann ebenfalls vom Web-Server geladen wird und in der Java Virtual Machine (JVM) des Browsers abläuft. Dieses Applet kommuniziert nun mit einem Admin-Server im Modul, der dann die Konfiguration des eingebetteten Systems über eine adäquate Schnittstelle vornimmt. Darüber hinaus kann der Admin-Server auch als SNMP-Agent für dieses eingebettete System fungieren.

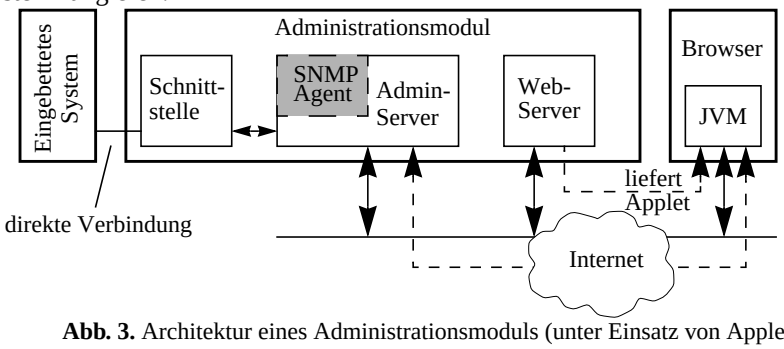

Abb. 3. Architektur eines Administrationsmoduls (unter Einsatz von Applets)

Die Realisierung der Administrationsfunktionalität in einem separaten Modul bietet drei essentielle Vorteile:

1. *Entflechtung der Entwurfskomplexität.* Da das eigentliche eingebettete System und das Administrationsmodul nur über eine (in der Regel einfache) Schnittstelle mit-

einander kommunizieren, verringert sich die Entwurfskomplexität des Gesamtsystems gegenüber einer integrierten Lösung, bei der eigentliche Funktionalität und Administration stärker miteinander wechselwirken.
2. *Bessere Auslegung der Teilsysteme.* Die Anforderungen an Rechenleistung und Speicherplatz können für beide Teilsysteme genauer bestimmt werden, als bei einer integrierten Lösung, da die zu implementierende Funktionalität der Teilsysteme schärfer umrissen ist. Infolgedessen können die Teilsysteme auch genauer den Bedürfnissen angepaßt werden.
3. *Nachrüstung des Administrationsmoduls in bereits existierende Systeme.* Um dem Wunsch nach Produktinnovation nachzukommen und die Bedürfnisse des Marktes abzudecken, wird es in den nächsten Jahren sehr häufig notwendig sein, internetbasierte Administration in bereits existierende Systeme nachträglich zu integrieren.

Zur Realisierung der Administrationsfunktionalität kann z.B. ein *„Embedded PC“* verwendet werden. Dabei handelt es sich um miniaturisierte PCs, die auf sehr engem Raum alle Funktionen zusammenfassen, die für einen Standalone-Betrieb erforderlich sind. Festplatten und Grafikkontroller sind hierbei häufig optionale Bestandteile, auf die verzichtet werden kann. Statt der Festplatte wird in der Regel eine Flash-Disk zum persistenten Speichern von Betriebssystem und Anwendung verwendet [2]. Die meisten Embedded PCs sind mit einer PC104-Schnittstelle ausgerüstet, über die es möglich ist, eine Vielzahl von Schnittstellen nachzurüsten (ISDN, Profibus, Interbus-S, etc.), um so das eigentlich zu administrierende System anzuschließen. Alternativ können spezielle Java-Prozessoren (picoJava, Patriot PSC1000) zur kostengünstigen Realisierung solcher Administrationsmodule verwendet werden.

4.1 Anforderungen an die Komponenten

Im folgenden sollen einige Komponenten dieser Architektur betrachtet werden, um die Anforderungen zu charakterisieren, die an sie gestellt werden.

Prozessor. Im Grunde werden an die Leistungsfähigkeit des Prozessors in dieser Architektur keine hohen Anforderungen gestellt, da er bei separaten Modulen nur für die im Rahmen der Administration anfallenden Aufgaben zuständig ist. Allerdings gilt es zu bedenken, daß der Prozessor bei vollständiger Verschlüsselung des Datenverkehrs zwischen Applet und Admin-Server nicht unerheblich belastet wird. Quantitative Aussagen kann man hierzu alledings nur mit Bezug auf konkrete Algorithmen machen, da sich die verschiedenen Verfahren im Rechenaufwand stark unterscheiden.

Selbstüberwachung. Der Embedded PC sollte über eine Selbstüberwachung, z.B. in Form eines Watchdog-Timers, verfügen, damit eventuelle Software-Fehler erkannt werden und das System nach einem Neustart dann wieder zur Verfügung steht. Auch diese Hardware-Komponente ist in den meisten kommerziell verfügbaren PCs vorhanden oder nachrüstbar.

Betriebssystem. Um die Integrität des Embedded PCs zu wahren und keine Angriffsfläche für potentielle Eindringlinge zu bieten, sollten die Netzwerkdienste des verwendeten Betriebssystems vollständig abgeschaltet werden (natürlich mit der Ausnahme des Web- und des Admin-Servers).

Admin-Server. Besondere Beachtung bei der Implementierung des Admin-Servers muß der Ansteuerung des zu administrierenden Systems geschenkt werden. Im Falle einer einfachen seriellen Leitung, wie sie in vielen Fällen ausreichend sein wird, können

die Schnittstellen des PCs über standardisierte Klassenbibliotheken verwendet werden (siehe auch [9]). Sollen spezielle Schnittstellen verwendet werden, dann kann entweder das *Java Native Interface* eingesetzt werden oder man überläßt die Ansteuerung einem gesondertem Prozeß (der z.B. in C programmiert werden kann), der mit dem Admin-Server über einen lokalen Socket kommuniziert.

5 Vorteile des Einsatzes von Java

Herkömmliche eingebettete Systeme werden häufig über serielle oder parallele Schnittstellen administriert. In diesen Fällen ist es in der Regel erforderlich, spezielle Software auf dem Rechner zu installieren, von dem aus die Administration vorgenommen werden soll. Der Einsatz von Java und insbesondere die Einbettung in das World Wide Web in Form von Applets bieten einige wesentliche Vorteile gegenüber diesen Techniken. Im einzelnen lassen sich diese Vorteile den Bereichen *Entwicklung, Kommunikation, Bedienerführung* und *Wartung* zuordnen.

Software-Entwicklung. Einige Vorteile von Java im Bereich der Software-Entwicklung sind mittlerweile allgemein bekannt. Dazu zählt zum einen die Möglichkeit, praktisch auf einer beliebigen Entwicklungsplattform arbeiten zu können, ohne Rücksicht auf das Zielsystem nehmen zu müssen. Zum anderen ist in Java bereits eine grafische Benutzungsoberfläche integriert, so daß auch in diesem Bereich keine Anpassung an das Zielsystem erforderlich ist.

Von besonderem Interesse bei der Entwicklung von Applets für die Administration und das Management von eingebetteten Systemen sind aber noch weitere Eigenschaften und Möglichkeiten von Java. Einen wichtigen Bereich stellen hierbei die Datenbankanbindungen dar, die in Java über vorhandene Klassenbibliotheken realisiert werden und so vom Entwickler abstrahiert werden können. Aber auch die Anbindung an objektorientierte Systeme wie CORBA, DCOM oder RMI stehen zur Verfügung. Für den Entwurf und die Realisierung von Add-On Lösungen zur Integration in bereits existierende Systeme ist auch die „Java Serial Port Standard Extension“ erwähnenswert, da sie die Ansteuerung der seriellen Schnittstelle unabhängig von der zugrundeliegenden Plattform ermöglicht.

Kommunikation. Reale Systeme sind häufig vor dem direkten Zugriff aus dem Internet durch eine Firewall geschützt, so daß eine Kommunikation zu dem zu administrierenden System nicht unbeschränkt durchgeführt werden kann. Das Java RMI-Konzept wurde bereits in Hinblick auf diese Problematik entwickelt und stellt verschiedene Methoden bereit, um eine solche Barriere zu überwinden. Darüber hinaus gibt es auch die Möglichkeit, ein Applet zu signieren und ihm damit die Rechte einzuräumen, vom Server aus angesprochen zu werden (Server-Socket). Eine weitere Betrachtung dieser Problematik ist in [3] zu finden.

Bedienerführung. Die Zahl eingebetteter Systeme, die uns umgeben, wird in den nächsten Jahren dramatisch ansteigen. Allein deswegen ist es vermutlich schon nicht möglich, Administration und Management dieser Systeme ausschließlich von geschultem Personal durchführen zu lassen. Statt dessen sollte die Bedienoberfläche auch von Laien benutzbar sein. Die Verwendung von WWW-Browsern und Applets bietet hierfür einen guten Ansatz, da diese Werkzeuge sehr weite Verbreitung finden und auch Laien mittlerweile geläufig sind. Außerdem bietet die Verwendung von Applets noch die Möglichkeit, Eingaben des Benutzers frühzeitig zu prüfen und kontextsensitive Eingabemasken zu erzeugen.

Wartung. Dadurch, daß sowohl der Server-Prozeß als auch das Applet auf dem selben Rechner gespeichert werden, wird auch die Wartung der Administrationssoftware selber erleichtert, weil nur an einer zentralen Stelle ein Update durchgeführt werden muß.

6 Anforderungen an ein Sicherheitssystem

Unter Sicherheit eines Systems wird häufig verstanden, daß alle Personen nur die ihnen erlaubten Aktionen ausführen dürfen. Es gibt aber vielfältige Aspekte, die den Grad an Sicherheit eines Systems beschreiben:

- **Authentifizierung** - Erst nach seiner eindeutigen Identifizierung, z.B. durch Eingabe eines Paßworts, darf ein Anwender das System nutzen.
- **Autorisierung** - Ein Anwender darf einen Dienst nur nutzen, wenn er nach geklärter Identität auch das Recht dazu hat.
- **Vertraulichkeit** - Das Abhören von Kommunikation sowie das Lesen oder Verändern fremder Daten darf nicht möglich sein.
- **Beherrschung** - Es müssen an allen internen und externen Schnittstellen Sicherungen integriert werden, um das System vor Angriffen schützen zu können.
- **Protokollierung** - Um Funktionsstörungen oder Angriffe erkennen und beseitigen zu können, ist ein Mechanismus zur Alarmierung und umfassenden Informierung nötig.
- **sichergestellte Beteiligung** - Jede Beteiligung eines Nutzers an einer Aktion muß zweifelsfrei nachgewiesen werden können.

Jedes Sicherheitskonzept muß diese Anforderungen erfüllen können, um als sicher zu gelten. Dabei wird es keine völlige Sicherheit geben, da Aufwand und Nutzen stets in einem akzeptablen Verhältnis bleiben müssen.

6.1 Existierende Modelle

Es gibt bereits Sicherheitskonzepte, die zum Schutz von Embedded Systems Anwendung finden können. Dazu zählen z.B. das bei WWW-Anwendungen oft eingesetzte Secure Hypertext Transfer Protocol (S-HTTP) oder Secure Socket Layer (SSL) sowie das Call-Back-Verfahren oder das Transaktionsnummernkonzept bei Client-Server-Modellen. Auch die Programmiersprache Java bietet zahlreiche Möglichkeiten.

WWW-Security. *S-HTTP* stellt eine Erweiterung des Hypertext Transfer Protocols um neue Header, HTML-Tags, Hyperlink-Anchor-Attribute und HTTP-Methoden dar [10]. Es bietet folgende Sicherungsmechanismen: Geheimhaltung durch Verschlüsselung, Authentifizierung durch digitale Zertifikate, Integrität bzw. Kommunikationsnachweis durch digitale Unterschriften. Server und Browser verhandeln vor dem Verbindungsaufbau darüber, welche dieser Maßnahmen eingesetzt und welche Verschlüsselungsalgorithmen verwendet werden sollen. Man kann also eine Nachricht verschlüsseln oder mit einer digitalen Unterschrift versehen und dann in eine S-HTTP-Nachricht einkapseln. Die Header dieses Paketes geben dem Empfänger die notwendigen Informationen, um die ursprüngliche HTTP-Nachricht zurückzugewinnen. Es gibt auch die Möglichkeit, Schlüssel für die Chiffrierung auf externem Weg (z.B. per Mail, Post, Telefon) auszutauschen und zur Decodierung eines erhaltenen S-HTTP-Dokuments einzusetzen.

Das Basisprotokoll für solche Sicherungen ist SSL (Secure Socket Layer). Es wird auf die existierende Transportschicht (z.B. TCP) aufgesetzt und besteht aus zwei Teilen: Das SSL Handshake Protocol behandelt den Verbindungsaufbau, d.h. die gegenseitige Authentifizierung, die Wahl des Verschlüsselungsverfahrens und den Austausch der Schlüssel [1]. Das SSL Record Protocol definiert ähnlich wie das IP Secure Protocol die Struktur der zu übertragenden verschlüsselten Daten sowie der dazugehörigen digitalen Unterschrift. Die gesamte Kommunikation erfolgt verschlüsselt.

Grundlage hierfür sind *kryptographische Algorithmen*, die nach der Art der verwendeten Schlüssel in asymmetrische und symmetrische Verfahren unterteilt werden können. Zu den bekanntesten Vertretern der symmetrischen Algorithmen (derselbe Schlüssel wird für Ver- und Entschlüsselung verwendet) zählt der Data Encryption standard DES, der in letzter Zeit häufig durch geglückte Angriffe Schlagzeilen machte. Sicherer, aber auch langsamer sind asymmetrische Verfahren (verwenden verschiedene Schlüssel) wie RSA, benannt nach seinen Erfindern Rivest, Shamir und Adleman. Mit ihrer Hilfe lassen sich digitale Signaturen erzeugen, die die Unveränderbarkeit von Daten garantieren.

Java. Das Sicherheitsmodell von Java unterscheidet zwischen *Applets*, die über das Internet geladen wurden, und lokalen Applikationen. Hierbei findet das sogenannte Sandkasten-Modell Anwendung: Aus dem Sandkasten heraus sind keine Zugriffe auf das lokale Dateisystem sowie kein Laden von lokalen Programmen oder Bibliotheken möglich, es dürfen nicht uneingeschränkt RMI-Verbindungen aufgebaut werden usw. Nur lokale Applikationen und vertrauenswürdige, d.h. signierte Applets, dürfen über den Sandkasten hinaus arbeiten.

Ein *signiertes Applet* wird auf dem Server-Host erzeugt. Dazu wird ein Java-Archiv (JAR - ähnlich Tar-File) erstellt und über einen geheimen Schlüssel mit einer digitalen Unterschrift versehen, die nur mit dem passenden frei zugänglichen Schlüssel verifizierbar ist. Dadurch kann garantiert werden, daß Applets nicht unerkannt von dritter Seite ausgetauscht werden. Es ist im Browser einstellbar, welchen Zertifizierungsstellen (die diese Schlüssel verwalten) vertraut werden soll.

Zur Absicherung der *RMI-Kommunikation* ist Java so konfiguriert, daß Applets außer zum lokalen und zu ihrem Ursprungs-Host keine Verbindungen aufbauen dürfen. Das bedeutet, daß Angreifer nicht in Kontakt mit dem Applet treten bzw. von ihm kontaktiert werden können. Auch bei Java-Applets können dabei kryptographische Techniken implementiert werden, die in der *Java Cryptography Architecture* (JCA) und im Java Cryptography Environment (JCE) definiert sind.

Zusätzlich mögliche Sicherheitsmaßnahmen. Beim sogenannten Call-Back-Verfahren wird die eigentliche Arbeitsverbindung nicht von den Clients, sondern durch den Server aufgebaut. Dadurch kann abgesichert werden, daß keine unautorisierten Zugriffe auf die vom Server angebotenen Dienste erfolgen, denn der Server verwaltet selbst eine Liste der autorisierten Clients und der ihnen erlaubten Aktionen. Ebenso können sogenannte *Transaktionsnummern* vereinbart werden. Dies sind einmalig verwendbare Paßwörter, die nacheinander bei jeder Transaktion verbraucht werden und Angriffe Dritter verhindern, sofern diese keine Kenntnis über die zuvor festgelegten Geheimnisse haben. Allerdings wird dadurch eine regelmäßige Vor-Ort-Administration notwendig, wenn die Transaktionsnummern aufgebraucht sind. Bei den heutigen Kapazitäten und Preisen für Speichermedien kann die Menge der speicherbaren Transaktionsnummern aber sehr groß und der Verwaltungsaufwand daher sehr klein sein.

Weiterhin sind zusätzliche softwareseitige Sicherungen denkbar, wie z.B. ein Transaktionskonzept mit Einzelbestätigung der erfolgten Aktionen. Die Verbindungen zwischen Server und Clients können abreißen, aber der Administrator muß zu jedem Zeitpunkt den genauen Zustand des Systems kennen, weshalb die Ausführung aller Aktionen durch die Clients bestätigt werden sollte.

7 Zusammenfassung

In diesem Beitrag wurde ein Einblick in den grundsätzlichen Stand der Remote-Administration von (eingebetteten) Systemen und den hierzu verwendbaren Basistechniken gegeben. In diesem Zusammenhang wurde ein neuartiges, allgemeines Add-On-Modell vorgestellt, das beschreibt, wie eingebettete Systeme für die IP-basierte Fernadministration in einer möglichst generischen Form erweitert werden können. Dabei wurde eine Unterscheidung zwischen *integrierten* und *modularen* Systemen getroffen. Durch einen modularen Aufbau und eine organisatorische Trennung der Fernadministration und -überwachung vom eigentlichen eingebetteten System werden dabei erhebliche Vorteile gewonnen: Innovationen auf dem Gebiet der sicheren Zugangstechniken können unabhängig von der für das eingebettete System erforderlichen Plattform realisiert werden; die Systeme sind voneinander entflechtet und lediglich über eine (plattformunabhängige) Schnittstelle miteinander verbunden. Über separate Administrationsmodule können so auch bereits existierende eingebettete Systeme um die zusätzliche Funktionalität der Fernadministration erweitert werden. Gleichzeitig können Management-Standards aus der Welt der EDV-Systeme (wie z.B. SNMP) übernommen werden, wenn das gewünscht wird. In einem eigenen Abschnitt wurden die Vorteile des Einsatzes von Java und die einsetzbaren Sicherheitstechniken diskutiert. Beispiele aktueller Anwendungsgebiete finden sich zahlreich: Telefonanlagen, Fahrkartenverkaufsautomaten für den öffentlichen Personennahverkehr, Fotokopiersysteme etc.

8 Literatur

1. Ahuja, V.: Network & Internet Security, AP Professional, Boston 1996
2. Blank, Hans-Joachim: "Embedded PCs: Applikation mit Intel-Prozessoren und Echtzeitsystemen", Markt und Technik Verlag (Design Elektronik) 1998
3. Falk, Rainer: Java RMI, CORBA und Firewalls, JIT’98, Springer Verlag 1998, S. 215-223
4. Hergenhan; Weiler; Weiß; Rosenstiel: Internet-basierte eingebettete Systeme in der industriellen Automation, GI/ITG-Workshop Java und eingebettete Systeme, Karlsruhe 1998, S. 19-31
5. Janssen, Rainer; Schott, Wolfgang: SNMP: Konzepte, Verfahren, Plattformen; DATACOM-Buchverlag 1993
6. N.N.: Beyond the Embedded Web Server, A Rapid Logic White Paper, December 1998, Rapid Logic Inc., http:\\www.rapidlogic.com
7. N.N.: Embedded Architecture for Web-Based Management, A Rapid Logic White Paper, May 1999, Rapid Logic Inc., http:\\www.rapidlogic.com
8. N.N.: Java Dynamic Management Kit - A White Paper, Sun Microsystems Inc., http://www.sun.com/software/java-dynamic/product.html
9. N.N.: Java Serial Port (COMM) Standard Extension, Sun Microsystems Inc., http://java.sun.com/products/javacomm/index.html
10. Nusser, St.: Sicherheitskonzepte im WWW, Springer-Verlag, Berlin 1998