

Ein generisches System für das Beobachten und Bedienen von SPS-Anwendungen auf der Basis von Web-Technologie

K.-P. Hermes^{2,1}, R. Kröger¹, M. Wack², P. Stammerjohann²

¹Fachhochschule Wiesbaden, Fachbereich Informatik
Labor für Verteilte Systeme
Kurt-Schumacher-Ring 18, D-65197 Wiesbaden
kroeger@informatik.fh-wiesbaden.de

²Moeller GmbH, Geschäftsbereich Automatisierung
Hein-Moeller-Straße 7-11, D-53115 Bonn
{K.Hermes, M.Wack, P.Stammerjohann}@moeller.net

Abstract. Der Beitrag stellt ein Web-basierten Systemansatz zum Bedienen und Beobachten von SPS-Anwendungen vor, der in einer Zusammenarbeit zwischen der Moeller GmbH und der Fachhochschule Wiesbaden entwickelt wurde. Ohne Programmierung auf Kundenseite können durch einfache Konfigurationsschritte beliebige Prozeßgrößen aus IEC 1131-konformen SPS-Anwendungen durch einen Web-Browser in frei konfigurierbarer Weise dargestellt und beeinflußt werden. Die beobachtete Performance des realisierten Prototyps war in allen beobachteten Fällen einem konventionellen Visualisierungssystem überlegen.

1 Einleitung

Die Integration von Speicherprogrammierbaren Steuerungen (SPSen) in Netzwerke gewinnt in zunehmendem Maße an Bedeutung. Eine konsequente Dezentralisierung von Automatisierungsstrukturen entspricht dem Trend, Intelligenz dort anzusiedeln, wo diese benötigt wird. Zusätzlich gibt es vermehrt Sensoren und Aktoren mit integrierter Intelligenz. Hieraus ergeben sich neue zusätzliche Anforderungen an die Vernetzung sowie die Kommunikation von SPS-Systemen untereinander. Aber auch in vertikaler Richtung, also von einer SPS zu Leitrechnern und in Richtung übergeordneter Geschäftsprozesse nimmt der Kommunikationsbedarf ständig zu.

Ein von einzelnen Herstellern unabhängiger Standard beseitigt viele der sich bei der Integration stellenden Probleme. Zudem werden die Aufwendungen zur Weiterentwicklung der Protokolle und Standards weltweit getragen. Für die oberen Netzwerkebenen der Automatisierungspyramide hat sich die auf TCP/IP basierende Internet-Protokollfamilie weltweit durchgesetzt. Sie bietet eine einfache, preiswerte und durchgängige Anbindung, die von allen gängigen Betriebssystemen unterstützt wird. Auch der Übergang vom Intranet zum weltumspannenden Internet ist, einmal von Firewalls abgesehen, ohne zusätzlichen Aufwand erreichbar. Die Anbindung von

SPSen an TCP/IP-basierte Netzwerke ist damit sehr attraktiv und eröffnet gleichzeitig neue Nutzungsmöglichkeiten.

Heute eingesetzte Anzeige- und Bedienschnittstellen beruhen häufig auf speziellen graphischen HMI-Bibliotheken mit wachsender Bedeutung von standardisierten Oberflächen wie Microsoft Windows. Dieser Trend wird insbesondere durch sogenannte Industrie-PCs gefördert, welche vor allem Datenverwaltungs- und Visualisierungsaufgaben übernehmen. Ein Beispiel hierfür ist der OPC-Ansatz [1], welcher einem Visualisierungsprogramm einheitliche Methoden zum Lesen und Schreiben von Variablen aus einem prozeßbezogenen Endgerät zur Verfügung stellt. Die notwendige Kopplung vom PC zum Endgerät muß jedoch i.d.R. vom Hersteller des Endgerätes auf Basis der existierenden Möglichkeiten realisiert werden, wie Feldbusse oder RS232-Schnittstellen mit darüberliegendem Protokoll.

Auf der anderen Seite bietet die auf der Internet-Protokollfamilie aufbauende Webtechnologie vielfältige attraktive Visualisierungsmöglichkeiten. In einer Zusammenarbeit zwischen der Moeller GmbH und der Fachhochschule Wiesbaden wurde untersucht, wie tragfähig ein Systemansatz zum Bedienen und Beobachten von SPS-Anwendungen auf der Basis von Webtechnologie ist [5]. Ziel war es dabei, einen generischen Ansatz zu verfolgen, so daß auf Kundenseite durch einfache Konfigurationsschritte beliebige Prozeßgrößen aus IEC 1131-konformen SPS-Anwendungen [2], [3] durch einen Web-Browser in frei konfigurierbarer Weise dargestellt werden können.

Das vorliegende Papier stellt den gewählten Ansatz vor. Das folgende Kapitel 2 stellt ausgehend von der generellen Struktur Web-basierter Anwendungen die wesentlichen Randbedingungen für eine Web-Integration von SPSen vor. Anschließend werden in den Kapiteln 3 und 4 Hardware- und Software-Architektur sowie die Realisierung des entwickelten Prototypen beschrieben. Kapitel 5 enthält eine Bewertung des Ansatzes, der Beitrag schließt mit Zusammenfassung und Ausblick in Kapitel 6.

2 Web-Technologie

Das World Wide Web hat in den vergangenen Jahren einen Siegeszug sondergleichen erlebt. Auf eine ausführliche Darstellung der Grundlagen Web-basierter Anwendungen soll hier verzichtet werden. Einen guten Überblick liefert [4].

Einem Web-basierten Systemansatz liegt zunächst ein TCP/IP-Netzwerk (Internet/Intranet) zugrunde. Web-Anwendungen sind Client/Server-orientiert, die Kommunikation zwischen Client und Web-Server wird durch das textbasierte HTTP-Protokoll geregelt, das TCP nutzt. Durch URLs identifizierte, übermittelte Dokumente beinhalten i.d.R. sogenannte Tags der Auszeichnungssprache HTML zur Festlegung der Dokumentenstruktur sowie vermehrt zur Festlegung der graphischen Repräsentierung. Browser sind die verbreitetsten Web-Clients und haben durch ihre einfache Bedienbarkeit und die Eigenschaft, die flexibel gestaltbaren HTML-Dokumente graphisch ansprechend darzustellen, eine hohe Akzeptanz erfahren und werden vielfach schon als universelle Benutzerschnittstellen angesehen. Neben der Möglichkeit, durch sogenannte Plugins (auf Client-Seite gespeicherte, dynamisch

ladbare Module) die Funktionalität des Browsers zur Darstellung neuer Medientypen zu erweitern, kann durch die heute übliche Integration einer Java Virtual Machine (JVM) beliebiger, in HTML-Dokumente als Applets eingebetteter, plattform-unabhängiger Anwendungscode mittels HTTP über das Netzwerk nachgeladen und ausgeführt werden. Java Applets eignen sich sehr gut, um komplexe interaktive Benutzeroberflächen zu implementieren. Das Applet-Sicherheitsmodell erlaubt es einem Applet, Netzwerkverbindungen zu Anwendungskomponenten auf dem Host herzustellen, von dem es (über einen dort laufenden Web-Server) ausgeliefert wurde. Über diese Netzwerkverbindungen können beliebige anwendungsspezifische Protokolle abgewickelt werden. Web-Server dienen der Bereitstellung von Dokumenten. Die für allgemeine Anwendungen genutzten Web-Server bieten neben der einfachen Auslieferung von statischen, in Dateisystemen gespeicherten Dokumenten verschiedene Möglichkeiten, angefragte Dokumente dynamisch zu generieren. Klassische Formen sind Server-Side Includes (SSI), die im Parsing des auszuliefernden Dokuments und der Ersetzung spezieller Markierungen durch häufig dynamisch generierte Daten auf Seiten des Servers bestehen. Außerdem besteht die Möglichkeit der Inanspruchnahme externer Anwendungen, die über das Common Gateway Interface (CGI) an den Web-Server angekoppelt sind. Hierbei wird einem i.d.R. für die Anfrage erzeugten Prozeß eine Eingabe des anfragenden Clients (z.B. der Inhalt eines ausgefüllten Formulars) übergeben; die Bearbeitung erzeugt unter anderem ein HTML-Dokument als Ausgabe, die das Gateway des Servers entgegennimmt und unverändert an den anfragenden Client ausliefert. Neuere Realisierungen dienen der Reduktion des mit CGI verbundenen Aufwands sowie der weiteren Flexibilisierung (z.B. Servlets).

Embedded Web-Server sind für kleine, i.d.R. Mikrocontroller-basierte Systeme ausgelegte Implementierungen des HTTP-Protokolls mit eingeschränkter Funktionalität gegenüber den beschriebenen Host-basierten Web-Servern und dadurch bedingtem geringen Ressourcenbedarf. Mittlerweile existieren am Markt eine Reihe von Embedded Web-Servern für verschiedene, in eingebetteten Systemen verwendete Betriebssystemkerne.

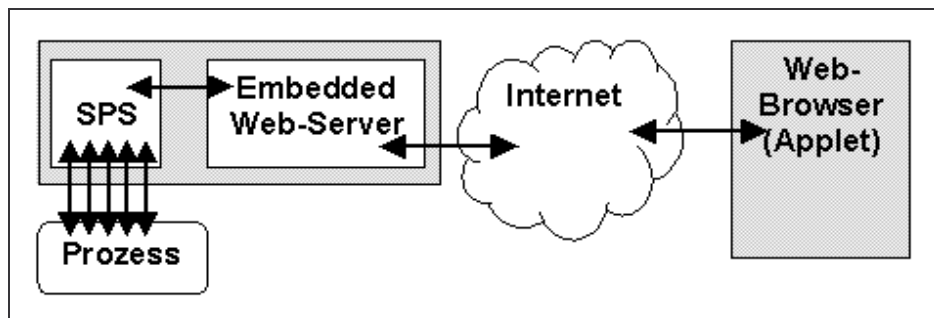


Fig. 1. Komponenten einer Web-basierten SPS-Anwendung

Ein Web-basierter Systemansatz für das Beobachten und Bedienen von SPS-Anwendungen verwendet prinzipiell die gleichen wie die oben beschriebenen Mechanismen. Auf Bedienerseite wird lediglich von einem üblichen Java-fähigen Web-Browser auf einem Desktop-System ausgegangen. Die Server-Seite wird durch einen mit der SPS verbundenen Embedded Web-Server gebildet. Notwendig ist, daß die eigentliche SPS-Anwendung in ihrem zeitlichen Verhalten nicht beeinträchtigt wird. Aufgrund der relativ schwachen CPU-Leistung heutiger SPSen verbietet sich eine Ausführung des TCP/IP-Stacks sowie des Embedded Web-Servers auf der CPU der SPS derzeit von vornherein. Vielmehr muß von der Bereitstellung zusätzlicher Ressourcen ausgegangen werden, z.B. in Form einer Mikrocontroller-basierten Einsteckkarte. Es ist daher auch ein Design-Ziel, die Verarbeitungslast zur Laufzeit i.w. auf den Bedienrechner zu verlagern. Dies geschieht, indem der Embedded Web-Server an den Browser HTML-Seiten ausliefert, die eingebettete Applets zur Präsentation der Prozeßgrößen und zur Interaktion mit dem Bediener enthalten. Die Applets werden dabei auf der JVM des Browsers ausgeführt und involvieren die SPS-Seite nur, um die aktuellen Prozeßgrößen bzw. die geforderten Benutzeraktionen zu kommunizieren. Hierzu werden spezielle aufwandsarme Protokolle vorgesehen.

3 Architektur des Prototyps

Die Architektur des entwickelten Prototypen entspricht dem in Abb. 1 dargestellten Aufbau. Die interne Architektur der SPS-Seite wird zunächst konkretisiert (vgl. Abb. 2). Das eingesetzte SPS-System PS416 von der Moeller GmbH ist modular aufgebaut und besitzt einen proprietären Rückwandbus. In diesen Rückwandbus wird ein modifiziertes μ Controller-Board MiniEthB C167 der Fa. Hightec, Saarbrücken, eingesteckt. Dieses besitzt einen C167-Prozessor, 1 MB RAM, 512 KB Flash-ROM, ein Ethernet-Interface und einen nach außen geführten Daten/Adreßbus. Die Modifikation adaptiert den Daten/Adreßbus der Karte über ein spezielles ASIC an den Rückwandbus der SPS. Die Kommunikation zwischen dem μ Controller-Board und der SPS erfolgt speicherbasiert über ein Dual-Ported-RAM im ASIC, das im Adreßraum der SPS-CPU und der CPU der Zusatzkarte zugreifbar ist.

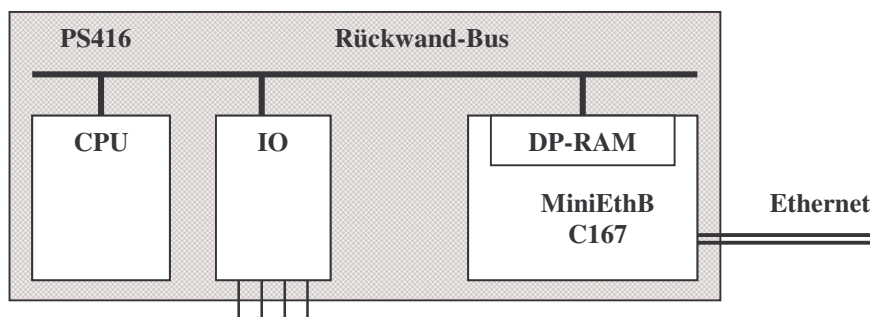


Fig. 2. Hardware-Struktur der SPS-Seite

Die Software-Seite unterscheidet zwischen dem Entwurfszeitpunkt der Beobachtungs- und Bedienoberfläche einer SPS-Anwendung sowie der Laufzeit des Systems, wenn die SPS-Anwendung Mithilfe der entworfenen Oberfläche bedient wird. Abb. 2 zeigt die verschiedenen Softwarekomponenten und deren Zusammenwirken zur Laufzeit.

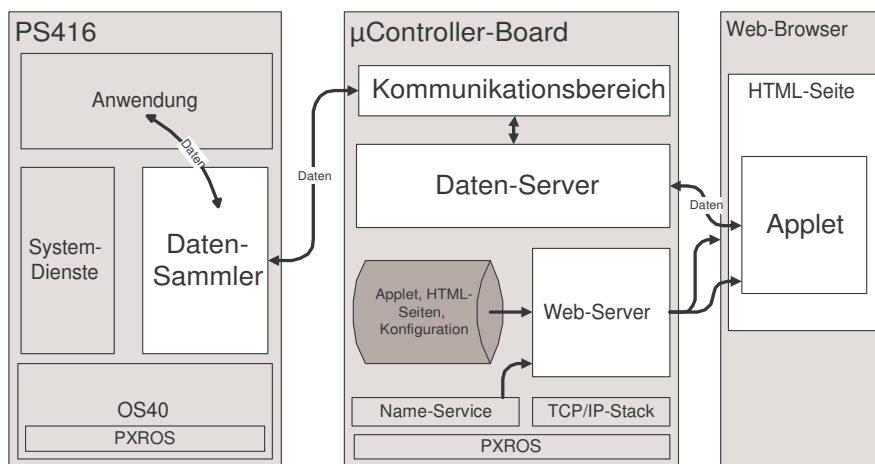


Fig. 2. Systemarchitektur

Zum Entwurf der Beobachtungs- und Bedienoberfläche und Erzeugung aller Daten für den Web-Server wird der sogenannte *Applet-Konfigurator* benutzt. Er leistet zunächst die Extraktion der Variableninformationen aus der IEC 1131-konformen SPS-Entwicklungsumgebung Moeller Sucosoft S40 [6]. Um die Sprachdefinition unverändert zu lassen, werden alle SPS-Programmvariablen, deren Namen ein bestimmtes Präfix besitzt, als potentiell für eine Visualisierung interessant angesehen. Für jede interessierende Variable wird automatisch eine eindeutige Variablennummer als später zu verwendende Kurzreferenz erzeugt sowie Offset und Länge der Variablen im Datenbereich der Funktionsbausteininstanz festgehalten.

Die Beobachtungs- und Bedienoberfläche kann ohne Programmierkenntnisse weitestgehend per Drag&Drop erstellt werden. Sie setzt sich aus elementaren Visualisierungskomponenten zusammen. Als grundlegende Typen sind z.B. ein analoges Zeigerinstrument, eine einzeilige und eine mehrzeilige Digitalanzeige, eine Binäranzeige, eine Balkenanzeige, ein Liniendiagramm, ein einfaches Textfeld, ein Schieberegler, ein Texteingabefeld und ein Button vorhanden. Der Applet-Editor als Komponente des Applet-Konfigurators bietet die Möglichkeit alle im System gefundenen Visualisierungskomponenten in Form von Grafiken anzuzeigen. Jeder Visualisierungskomponente wird je nach Typ eine oder mehrere der interessierenden Programmvariablen zugeordnet. Die Anzeige und Auswahl erfolgt in Form eines auffächerbaren Baumes. Visualisierungskomponenten können die Möglichkeit der Dateneingabe vorsehen. Sie sind damit Interaktionselemente, deren Eingabewert zur

Laufzeit an die jeweils zugeordnete Variable in der SPS über Daten-Server und Daten-Sammler weitergeleitet wird. Ein Arbeiten mit den realen Variableninformationen aus der laufenden SPS-Anwendung ist schon zum Entwurfszeitpunkt möglich, was das Erstellen von grafischen Anzeigen erleichtert, da so viel früher das spätere Erscheinungsbild beurteilt werden kann.

Nachdem eine Visualisierungsoberfläche erstellt wurde, kann sie mit allen nötigen Daten als *Applet* abgespeichert werden. Zusätzlich kann automatisch eine HTML-Seite generiert werden, in die das Applet mit der zugehörigen Konfiguration eingebettet wird. Das Applet stellt eine Untermenge der Java-Klassen des Applet-Konfigurators dar. Beim Abspeichern der Oberfläche aus dem Applet-Konfigurator kann der Anwender wählen, ob das Applet zur Laufzeit rekonfiguriert werden können soll oder ob die erzeugte Oberfläche in ihrer Struktur statisch ist. Die zu dem entsprechenden SPS-Projekt erzeugten Daten werden zusammen mit den SPS-Projekt in dem jeweiligen S40-Projektverzeichnis abgelegt. Schließlich kann ein Abbild erzeugt werden, das im Flash-RAM der Zusatzkarte als Datenteil des Web-Servers abgelegt wird.

Zur Laufzeit des Systems existieren die folgenden Komponenten:

- das Applet im Browser des Bedieners,
- ein Embedded Web-Server und ein sogenannter Daten-Server auf der μ Controller-basierten Zusatzkarte,
- ein Datensammler als SPS-Betriebssystemerweiterung mit geringen Betriebsmittelanforderungen.

Diese Aufteilung erreicht, daß in der SPS nur sehr wenig Rechenzeit aufgewendet werden muß, der μ Controller der Zusatzkarte für die Internet-Anbindung vollständig zur Verfügung steht und die hohe Rechenleistung des Bediener-PCs zur eigentlichen Erzeugung der graphischen Darstellung verwendet wird.

4 Realisierung

Der Applet-Konfigurator besteht aus dem Applet-Editor, dem Variablen-Editor und einem Konfigurationstool für den Embedded Web-Server. Alle Teile sind generisch und damit unabhängig von einer konkreten Anwendung realisiert. Der *Applet-Konfigurator* und die Visualisierungskomponenten wurden vollständig in der Programmiersprache Java implementiert. Im folgenden wird der Applet-Editor detaillierter beschrieben.

Sämtliche Informationen bezüglich einer Applet-Konfiguration werden zusammen mit dem Projekt der SucoSoft S40 in dem jeweiligen Projektverzeichnis abgespeichert. Weiterhin ist es möglich, mehrere Applets pro Projekt zu verwalten. So können spezielle Ansichten für einzelne Teilbereiche erstellt werden. Die von der Entwicklungsumgebung benötigten Variableninformationen werden immer aus dem zu letzt bearbeiteten S40-Projekt extrahiert. Bestandteil dieser Informationen sind neben der eindeutigen Variablennummer der Variablenname, eine sprechende Bezeichnung, der Wertebereich, Umrechnungsfaktoren für eine lineare Transformation, die Instanznummer, der Offset innerhalb einer Instanz, die Anzahl

der Bytes und die Übertragungsrate. Alleine durch die Informationen bezüglich Offset und Größe der Daten innerhalb einer Instanz sind die Programmteile auf dem μ Controller und der SPS zur Laufzeit in der Lage Daten aus der SPS-Anwendung zu liefern. Die Variableninformationen werden im Applet-Editor in einer Baumstruktur abgebildet. Enthalten sind dabei die Funktionsbausteininstanzen welche zu visualisierende Variablen enthalten, globalen Variablen und die benutzten Visualisierungskomponenten. In Abb. 3 ist der Applet-Editor mit einer erzeugten Applet-Oberfläche dargestellt.

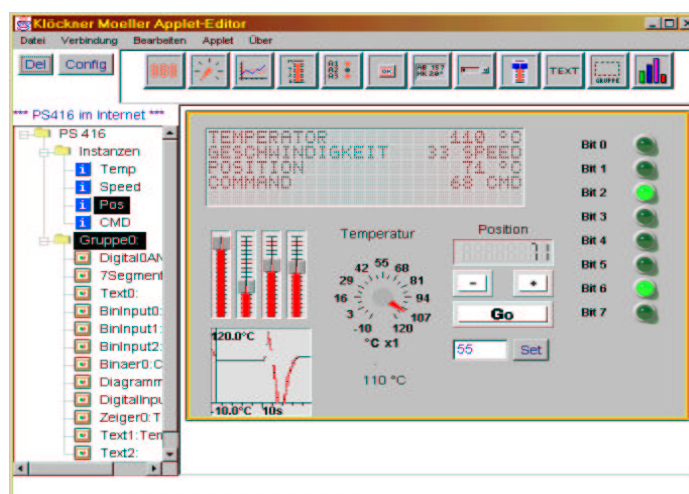


Fig. 3. Applet-Editor mit Beispielkonfiguration

Das Applet lädt beim Starten die vom Applet-Konfigurator generierten Informationen aus einer Datei, die vom Embedded Web-Server zur Verfügung gestellt wird. Damit ist das erzeugte Applet als auch der Applet-Editor jeweils in der Lage, eine Verbindung zum Daten-Server herzustellen, und so mit aktuellen Variablenwerten zu arbeiten. Die Kommunikation läuft in beiden Programmteilen über die gleiche Kommunikationsklasse ab. Die einzelnen Variableninhalte werden mittels einer Listenklasse, in der auch eine lineare Transformation der Werte vorgenommen werden kann, an die verschiedenen zugeordneten Visualisierungskomponenten verteilt. Nachdem ein vollständiges Prozessabbild empfangen worden ist, wird die grafische Darstellung der Visualisierungskomponenten aktualisiert. Die einzelnen Visualisierungskomponenten werden zur Laufzeit dynamisch eingebunden, ähnlich wie Plugins, wodurch das System sehr einfach um neue Komponenten erweiterbar ist.

Wenn ein Wert einer Visualisierungskomponente an die SPS übermittelt werden soll, sendet sie den Wert zusammen mit der eindeutigen Variablennummer an die Kommunikationsklasse, die wiederum mittels der Listenklasse eine lineare

Transformation durchführen kann. Der errechnete Wert wird anschließend durch die Kommunikationsklasse an den Daten-Server übermittelt.

Die Kommunikation zum Daten-Server wird über das HTTP-Protokoll getunnelt, indem die Verbindung durch ein Aufruf eines CGI-Scriptes auf dem Web-Server hergestellt wird, der die Verbindung an den Daten-Server weiterleitet. Neben dieser Funktionalität bietet der Web-Server auch Server-Side-Includes, mit Hilfe derer unter anderem auch aktuelle Variablenwerte in eine HTML-Seite eingebunden werden können.

Der *Daten-Server*, der in „C“ realisiert ist, besteht aus zwei Tasks, dem Daten-Service und dem Daten-Verteiler. Gemeinsam verwalten sie eine globale Liste, in der alle zu übermittelnden Variableninformationen gespeichert werden. Nachdem eine Verbindung vom Applet bzw. Applet-Editor zum Daten-Verteiler aufgebaut worden ist, werden alle benötigten Variablen mittels der eindeutigen Variablennummer und der Position im Speicher angefordert. Der Daten-Verteiler trägt alle angeforderten Variablen in die globale Liste ein. Anschließend abonniert der Daten-Service die neu eingetragenen Variablen bei der SPS. Beim Beenden eines Applets werden bestellte Werte auf die gleiche Weise wieder abbestellt. Der Daten-Service bekommt die Variableninhalte aller bestellter Variablen zusammen mit der eindeutigen Variablennummer von der SPS in zyklischen Abständen übermittelt. Er trägt die empfangenen Werte in die globale Liste ein, aus welcher der Daten-Verteiler anschließend die Variableninhalte zusammen mit der eindeutigen Nummer an die verschiedenen Applets verteilt. Auf die gleiche Weise erfolgt die Werteübermittlung von den Applets zur SPS. Der einzige Unterschied besteht darin, daß hier die Kommunikation mit mehreren Applets geregelt werden muß. So gibt es hier den Sonderfall, daß mehrere Applets einen Wert in die selbe Variable schreiben wollen, oder daß ein Applet kurz aufeinander verschiedene Werte in die selbe Variable schreiben will, was z.B. bei einem Tastendruck vorkommen kann. Hier wird die Übertragung der einzelnen Werte sichergestellt, indem in den jeweiligen Sonderfällen die Verarbeitung der von den Applets gesendeten Daten erst fortgesetzt wird, nachdem der jeweilige Wert übermittelt worden ist. Der Daten-Verteiler und der Daten-Service laufen immer abwechselnd, indem sie jeweils nacheinander auf die globale Liste zugreifen. Dieses Vorgehen hat sich als das performanteste herausgestellt, da ein Taskwechsel immer nur dann stattfindet, wenn entweder ein Prozessabbild komplett übertragen oder empfangen worden ist. Durch dieses Vorgehen wird weiterhin sichergestellt, daß immer ein vollständiges Prozessabbild übertragen wird, bevor neue Werte von der SPS bzw. einem Applet an die SPS übertragen werden können.

Der Daten-Service tauscht die Daten mit dem Daten-Sammler in der SPS über das Dual-Ported-RAM im ASIC aus. Dazu ist das DPR in zwei unterschiedlich große Puffer zum Lesen und Schreiben eingeteilt, auf die mittels gegenseitigen Ausschlusses zugegriffen werden kann. Auf diese Weise wird die Übertragung des Daten-Services mit der Datenübertragung des Daten-Sammlers synchronisiert. Innerhalb jedes Zyklus werden sämtliche beim Daten-Sammler angeforderten Variablen übertragen. Die unterschiedliche Puffergröße basiert auf der unterschiedlichen Menge der zu übertragenden Daten. So kann der Benutzer im Normalfall nicht so viele Daten über die Applet-Oberfläche eingeben, wie in der SPS anfallen, da der Anwender immer nur über eine Visualisierungskomponente einen

Wert eingeben kann, wohingegen in der SPS immer ein Prozessabbild übertragen werden muß, was deutlich mehr Werte beinhaltet.

Der *Daten-Sammler*, der in „C“ realisiert ist, läuft als eigenständiger Task im Betriebssystem der PS416. Somit kann diese unabhängig von der laufenden SPS-Anwendung auf deren Speicherinhalte zugreifen. Ähnlich wie beim Daten-Server verwaltet der Daten-Sammler eine Liste aller abonnierten Variablen, die zyklisch abgearbeitet wird. Dazu werden zuerst alle von dem Daten-Service empfangenen Werte in das RAM der SPS geschrieben. Anschließend werden die angeforderten Variablen in die Liste kopiert. Dieses Vorgehen gewährleistet, daß die Zugriffszeit auf die Variablen der laufenden Anwendung minimiert wird und eine Übermittlung eines konsistenten Prozessabbildes möglich wird. Nachdem alle Variablen aus der Anwendung ausgelesen worden sind, werden die in der Liste eingetragenen Werte an den Daten-Service übermittelt. Durch die beschriebene Verarbeitung der Werte ist eine sichere und optimale Übertragung der Werte zwischen den einzelnen Komponenten sichergestellt.

5 Bewertung

Mit Hilfe des vorgestellten Systems läßt sich die Visualisierung einer SPS-Anwendung in kürzester Zeit realisieren, weil insbesondere die heute noch vorhandene Trennung zwischen Programmierung und Visualisierung aufgehoben wird. Damit entfallen die sonst notwendigen und fehlerträchtigen Doppeleingaben für zu visualisierende Variablen. Weiterhin müssen an einer Anwendung keine semantischen Modifikationen vorgenommen werden. Damit sind auch existierende Applikationen darstellbar.

Zur Durchführung der Visualisierung ist weder eine spezielle Software noch eine spezielle Hardware notwendig. Es wird lediglich ein Web-Browser sowie ein Ethernet- bzw. Internet-Anschluß benötigt, welches beides auf heutigen PCs standardmäßig vorhanden ist. Insbesondere sind die eingesetzten Techniken sowohl in einem Firmennetz (Intranet) als auch weltweit (Internet) ohne Modifikationen einsetzbar.

Messungen in Hinblick auf die erzielbare Update-Rate von verschiedenen Views ergaben überraschenderweise, daß die Web-basierte Oberfläche einer vergleichbaren konventionellen Oberfläche sogar überlegen war.

Insgesamt konnte damit gezeigt werden, daß ein Web-basierter Ansatz für das Beobachten und Bedienen von SPS-Anwendungen tragfähig ist. Die Attraktivität des Ansatzes steigt dadurch weiter, daß die Integration von Audio- oder Bildausgaben, Animationen, usw. zu multimedialen Bedienoberflächen einschließlich entsprechenden Hilfesystemen ohne weiteren Aufwand möglich ist. Die Ausnutzung dieser Techniken erschließt einerseits neue Einsatzfelder für konventionelle SPSen, andererseits wird die Integration von PC und SPS-Technologie drastisch enger. Damit wird die nahtlose Integration von SPSen in übergeordnete, verteilte Geschäftsprozesse deutlich realistischer.

6 Zusammenfassung und Ausblick

In diesem Beitrag wurde ein generisches System zur Visualisierung von speicherprogrammierbaren Steuerungen auf Basis von Web-Technologie vorgestellt. Ausgehend von der heute nur ungenügend unterstützten Anbindung von SPS-Systemen in offene Kommunikationsstrukturen basiert die vorgestellte Vorgehensweise auf weltweit akzeptierten Standards und Technologien.

Es kann als mittlerweile gesichert gelten, daß sich TCP/IP als Protokollfamilie des Internets weltweit in kommerziellen Netzen durchgesetzt hat. Auch wenn für viele Anwender Internet und World Wide Web (WWW) gleichbedeutend sind, bleibt festzuhalten, daß neben dem Web-Protokoll HTTP weitere wichtige Internet-Protokolle wie SMTP für den Versand von Electronic Mail und SNMP als Management-Protokoll existieren und von zahlreichen Anwendungen eingesetzt werden.

Die durch die Ergebnisse der vorgestellten Arbeit erreichte Anbindung von Geräten und Teilsystemen in heutige TCP/IP-basierte Netzwerke kann für vielfältige Aufgaben der Konfiguration, Inspektion/Wartung, Qualitätssicherung, Maschinendatenerfassung, Produktionsplanung und –vorbereitung sowie zur Anbindung an kaufmännische Systeme eingesetzt werden. Insbesondere der Aspekt der Fernwartung und –diagnose kommt eine besondere Bedeutung zu.

Zusammenfassend läßt sich festhalten, daß der Machbarkeit nach den Erfahrungen nichts im Wege steht und das bei weiter fortschreitender Infrastruktur solche Anwendungen in der industriellen Praxis kurz vor der Einführung stehen.

7 Literaturverzeichnis

- [1] OPC Spezifikation; OPC foundation P.O. Box 140524, Austin, Texas , 1998
- [2] IEC1131-Standard, Part 1-5; International Electrotechnical Commission, Genf, 1992
- [3] Neumann, P.; Grötsch, E.E.; Lubkoll, C.; Simon, R.: "SPS-Standard IEC 1131 - Programmierung in verteilten Automatisierungssystemen", Oldenbourg-Verlag, 1995
- [4] Turau, V.: "Techniken zur Realisierung Web-basierter Anwendungen", Informatik Spektrum 22: 1-2 (1999), Springer-Verlag, 1999
- [5] Hermes, K.-P.: "Generische Visualisierung von SPS-Anwendungen auf Basis von Web-Technologie", Diplomarbeit an der FH-Wiesbaden, 1999
- [6] SUCOSOFT S40-Benutzeroberfläche, Moeller GmbH, 1998