Effizientes Testen heterogener Leitsystemkonfigurationen

Integration gewerkeübergreifender Hardware-Emulatoren

Die Simulation als Mittel zum Testen der Leitsystem-Konfiguration hat sich über die letzten Jahre etabliert. Aufgrund des hohen manuellen Aufwandes fokussieren simulationsbasierte Testmethoden meist auf den funktionalen Bereich, wobei ausschließlich die Steuerungslogik, nicht aber die Konfiguration der Subsysteme getestet wird. Der Beitrag stellt eine Methode vor, mit der die Vorteile simulationsbasierter Tests um die Möglichkeiten aktueller Hardware-Emulatoren erweitert werden. Hierzu wird ein Framework präsentiert, welches die für einen Subsystemtest notwendigen Emulatoren integriert und automatisch konfiguriert.

SCHLAGWÖRTER Emulation / Subsystemtest / Virtuelles Framework

Virtual Emulator Framework –
Domain independent Integration of Emulators

During the past several years, simulation has been established for testing process control system configuration. Because of the high manual effort for the development of simulation models, those usually focus on the functional area, whereas only the correctness of the central control logic, but not the configuration of the subsystems is tested. This contribution presents a method that extends the advantages of simulation by the possibility to use modern hardware emulation, as well. Thus, a framework is presented that integrates and automatically configures control system- and subsystem emulators in order to be able to perform comprehensive tests of automation systems.

KEYWORDS Emulation / subsystem test / Virtual Framework

MARIO HOERNICKE, JÜRGEN GREIFENEDER, MIKE BARTH, ABB Forschungszentrum

Im Engineering von Automatisierungssystemen (AT-System) für Anlagen der Prozessindustrie gewinnt die Phase des Testens zunehmend an Bedeutung. Je komplexer die Anlagen werden, desto umfangreichere Tests sind nötig, um die korrekte Funktion des AT-Systems im Sinne der Kundenspezifikation zu überprüfen. Angelehnt an die Testprozeduren von Software-Applikationen beinhaltet das Engineering eines Prozessleitsystems (PLS) mehrere Prüfschritte. Je nach Implementierungsfortschritt werden Verriegelungen, Schrittketten, Grenzwerte (für Schaltungen, Alarme und Meldungen), sowie Wirkrichtungen von Reglern daraufhin überprüft, ob sie den Spezifikationen entsprechen. Die Ingenieure stehen vor der Herausforderung, dass die für den Factory Acceptance Test (FAT) zur Verfügung stehende Zeit immer kürzer wird [1], wohingegen die Anzahl und Komplexität der durchzuführenden Tests aufgrund des zunehmenden Funktionsumfangs der Prozessanlage und damit auch des AT-Systems stetig zunimmt. Des Weiteren erhöht die geografische Verteilung der am Engineering beteiligten Ingenieure die Signifikanz des FAT als finale Integration und Qualitätssicherung des AT-Systems.

Der FAT wird jedoch durch das Fehlen von realen AT-Komponenten (zum Beispiel Feldbuskomponenten) im Prüffeld erschwert. Insbesondere Peripherie-Komponenten werden aus logistischen Gründen oftmals direkt zur Anlage versandt. Das Prüffeld besteht daher in der Regel aus Beispiel-Hardware-Komponenten, welche nur den Test von Teilsystemen erlauben [2]. Um das AT-System während des FAT dennoch so weit als möglich testen zu können, werden zunehmend Methoden der virtuellen Inbetriebnahme [3] eingesetzt. Dazu wird der zu automatisierende Prozess mittels geeigneter Werkzeuge simuliert. Die Steuerungsprogramme laufen dabei entweder auf der realen Hardware oder auf emulierten Steuerungen (Soft-SPS). Mit einer Soft-SPS lässt sich die Steuerungslogik auch ohne das Vorhandensein realer Hardware testen [4].

Um bei simulationsbasierten Tests nicht ausschließlich auf den funktionalen Bereich zu fokussieren, werden Emulatoren für die projektierten Subsysteme sowie deren Peripherie eingesetzt. Oft ist es die Peripherie des AT-Systems, welche den größten Aufwand im Rahmen der Testphasen – insbesondere während des FAT – erzeugt. Wie in Bild 1 dargestellt, existieren Emulatoren für einfachere E/A-Applikationen (E/A-Emulator) bis hin zu komplexen Emulatoren für Feldbussysteme. Beispiele sind Emulatoren für elektrische Verteilsysteme [5] auf Basis des IEC 61850 Standards [6] beziehungsweise Emulatoren für eine auf Foundation Fieldbus (FF) basierende dezentrale Steuerungslogik [7].

Da die Steuerungslogik zunehmend auf intelligente Feldgeräte verteilt wird [8], bedeutet der Einsatz von Hardware-Emulatoren für den Test des AT-Systems einen wichtigen Schritt in Richtung einer höheren Testabdeckung. In Anlehnung an [9] unterscheiden die Autoren dabei zwischen dem Einsatz von Emulation und Simulation. Emulation beschreibt die Abarbeitung einer Software auf einem Fremdsystem. Dabei muss sichergestellt sein, dass das Verhalten nach außen dem des Originalsystems entspricht. Auf ein AT-System übertragen bedeutet dies, dass ein Steuerungsprogramm beispielsweise nicht auf der realen SPS abgearbeitet wird, sondern auf einer Soft-SPS, welche die reale SPS emuliert. Diese Form wird dahingehend auch als Hardware-Emulation bezeichnet.

Simulation beschreibt hingegen nach [10] „ein Verfahren zur Nachbildung eines Systems mit seinen dynamischen Prozessen in einem experimentellen Modell …“. Wiederum auf den Bereich der Automatisierungstechnik übertragen, wird Simulation überwiegend für die Nachbildung der zu steuernden / regelnden Anlage eingesetzt.

Wie in Bild 1 dargestellt, berücksichtigt der in diesem Beitrag vorgestellte Ansatz nicht ausschließlich Soft-SPS-Emulatoren sondern darüber hinaus den Einsatz von komplexeren Hardware-Emulatoren für unterschiedliche Bussysteme. Die darüber hinausgehende Notwendigkeit der Prozesssimulation wird in [11] und [12] beschrieben.

Trotz der erhöhten Testabdeckung werden HW-Emulatoren bislang nur in begrenztem Umfang verwendet.

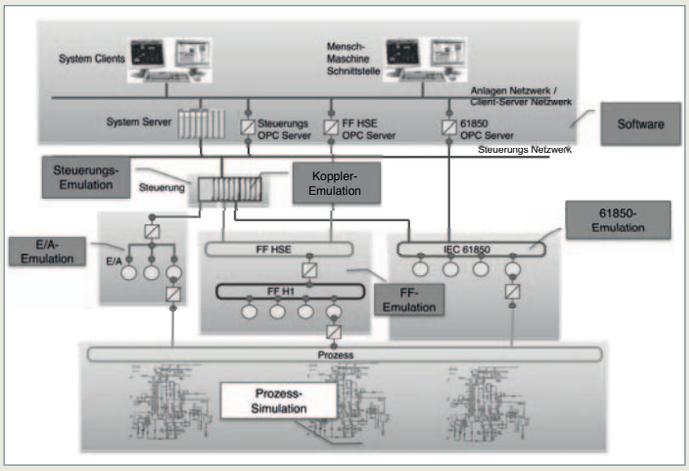

BILD 1: Virtuelle Hardware-in-the-Loop-Testumgebung

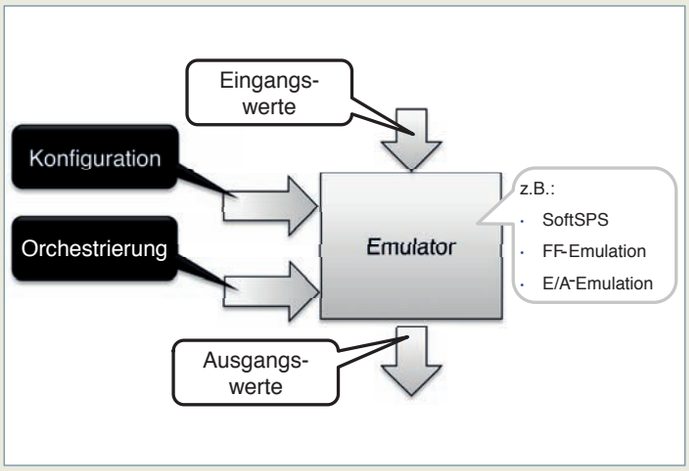

BILD 2: Funktionsweise eines Emulators

Begründet wird dies durch den hohen manuellen Aufwand zur Parametrierung und Konfiguration der Emulatoren. Des Weiteren wird ein beträchtlicher Aufwand für Bereitstellung, Administration und Konfiguration der benötigten IT-Infrastruktur notwendig. Um eine einheitliche und automatisierte Konfiguration – bestehend aus Emulatoren und umgebender IT-Infrastruktur – zu ermöglichen, wird das Konzept eines virtuellen Emulator Frameworks (VEF) vorgestellt.

1. EMULATOREN IN DER AUTOMATISIERUNGSTECHNIK

In Erweiterung der zuvor getroffenen Definitionen umfasst die Emulationstechnologie nach [9] die Nachbildung des Verhaltens eines Geräts mittels eines anderen Geräts oder einer Software. Dieser Ansatz wird in unterschiedlichen Gewerken verwendet, um Systemtests (zum Beispiel Funktions- und Integrationstests) durchführen zu können, bevor die realen Hardwarekomponenten zur Verfügung stehen. Zusätzlich werden Emulatoren für Feldbus-Systeme eingesetzt, um die Logik [7] sowie das zeitliche Verhalten des Busses [14] realitätsnah zu testen.

Die wachsende Vielfalt an Emulatoren erschwert jedoch deren effizienten Einsatz, da jeder Emulator individuell konfiguriert, parametriert und initialisiert werden muss. Bei großen Systemen mit verschiedenen Subsystemen und komplexer Peripherie müssen die Emulatoren exakt aufeinander abgestimmt sein. Oftmals entsteht durch die Konfiguration und Verknüpfung der Emulatoren ein Netzwerk, welches dem Komplexitätsniveau des realen AT-Systems nahekommt und in einem deutlichen Mehraufwand, im Vergleich zum Teilsystemtest, für den FAT resultiert.

Dabei ist der für die Ausführung der Emulation notwendige Aufwand zur Konfiguration der IT-Infrastruktur noch nicht berücksichtigt: Die meisten Emulatoren benötigen eine separate Ethernet-Schnittstelle mit einer auf den Emulator abgestimmten IP-Adresse. Einige Emulatoren sind zudem nicht multiinstanzfähig, weshalb mehrere Instanzen des gleichen Typs auf verschiedenen Rechnern verteilt emuliert werden müssen. Der hierfür notwendige manuelle Aufwand ist beträchtlich.

1.1 Konfiguration am Beispiel von FF

Die Funktionsweise von Emulatoren für AT-Komponenten basiert auf immer gleichen Prinzipien. Zunächst muss der Emulator die Konfiguration der zu emulierenden realen Komponente kennen. Dazu wird eine Konfiguration benötigt, welche auf eine der beiden nachfolgend beschriebenen Möglichkeiten aufgesetzt werden kann.

1 | Die Konfiguration eines einzelnen Geräts, zum Beispiel einer SPS, kann mittels eines Downloads der Applikation auf die Soft-SPS durchgeführt werden. Diese Variante wird dann eingesetzt, wenn ein separates Gerät durch eine Instanz des Emulators emuliert wird. Die Konfiguration basiert dabei auf den gleichen Kommunikationsprotokollen wie beim realen System.

2 | Eine Datei wird auf Basis von Daten des Engineering-Werkzeuges erzeugt und zur Konfiguration des Emulators verwendet. Diese Möglichkeit wird zur Konfiguration von Subsystemen verwendet, deren Topologie dem Emulator vor der Konfiguration nicht bekannt ist.

Ein Beispiel für einen Emulator mit dateibasierter Konfiguration ist der FF-Emulator nach [7]. Da eine FF-Lösung üblicherweise aus mehreren voneinander getrennten Subnetzen besteht, muss zunächst der Emulationsrahmen festgelegt werden. Der hierfür referenzierte Emulator ist in der Lage, in einer Instanz ein Subnetz zu emulieren. Dies bedeutet, dass der Nutzer zunächst das zu emulierende Subnetz auswählen muss.

Als nächstes wird die Konfiguration des ausgewählten Subnetzes benötigt. Hierfür wird eine Schnittstelle verwendet, welche auf die Daten des Engineering-Werkzeugs zugreift und diese in ein dateibasiertes Zwischenformat exportiert, welches sich problemlos auf andere Rechner übertragen lässt.

Da FF ein ethernetbasierter Feldbus ist, muss für jedes zu emulierende Subnetz eine Ethernet-Schnittstelle am Emulations-PC verfügbar sein oder gegebenenfalls nachgerüstet werden. Damit eine Verbindung zum umgebenden Leitsystem hergestellt werden kann, muss diese Ethernet-Schnittstelle konfiguriert werden (IP-Adresse, Subnetz-Maske und eventuell Gateway). Nun kann der Emulator gestartet und die zuvor erzeugte Konfigurationsdatei geladen werden. Die Konfiguration von weiteren Emulatoren wird analog zu der beschriebenen Vorgehensweise durchgeführt.

Für die Emulation des gesamten AT-Systems müssen die Emulator-Instanzen auf entsprechend unterschiedliche Rechner verteilt, die benötigte Hardware gegebenenfalls nachgerüstet sowie die Hardware und der Emulator selbst konfiguriert werden. Das in diesem Beitrag vorgestellte VEF verfolgt die Automatisierung des beschriebenen Vorgehens.

1.2 Orchestrierung

Nachdem die Emulation initialisiert wurde, liest der Emulator, wie in Bild 2 gezeigt, die an sein Subsystem gesendeten Eingangswerte ein, führt die Logik der zu emulierenden Komponente aus und schreibt die berechneten Ausgangswerte zurück an die übergeordnete AT-Komponente. Zur Kommunikation der Ein- und Ausgangswerte werden die spezifischen Protokolle (zum Beispiel Profibus) der zu emulierenden Komponente verwendet.

Einige Emulatoren stellen zusätzliche Steuerungsfunktionen zur Verfügung. Neben den Basisfunktionen Start und Stopp, kann der Nutzer aktuelle Zustände einfrieren, speichern und diese zu einem späteren Zeitpunkt wiederherstellen. In diesem Zusammenhang umfasst ein weiteres Ziel des VEF die Vereinheitlichung der notwendigen (Fern-) Orchestrierung der Emulatoren durch ein einheitliches Bedienkonzept. Da die Struktur verschiedener Emulatoren auf ähnlichen Prinzipien beruht, liegt es nahe, ein Konzept zur Abstraktion der beschriebenen Funktionalität zu entwerfen. Diese Abstraktion kann als Schnittstelle dienen und die Konfiguration und die zeitgleiche Steuerung mehrerer Emulator-Instanzen ermöglichen.

1.3 Anforderungen an das VEF

Das VEF soll die Administration der Rechnerinfrastruktur sowie den Entwurf der Emulation vereinfachen und automatisieren. Basierend auf dem zu erzielenden Nutzen, lassen sich vier grundlegende Anforderungen an das VEF bezüglich dessen Nutzung und Administration formulieren.

1 | **Automatische Konfiguration der Emulatoren**
Jeder Emulator benötigt die Konfiguration der zu emulierenden Komponente. Das VEF muss in der Lage sein, diese Konfiguration – mit Hilfe des integrierten Emulators – automatisch zu erzeugen und dem Emulator bekannt zu geben.

2 | **Automatische Konfiguration der IT-Infrastruktur**
Die IT-Infrastruktur muss auf die auszuführenden Emulatoren abgestimmt und entsprechend skalierbar sein, damit die benötigten Ressourcen automatisch bereitgestellt und die notwendige Konfiguration automatisiert durchgeführt werden kann.

3 | **Einheitliches Bedienkonzept**
Die Bedienoberfläche soll unabhängig von den integrierten Emulatoren sein. Ein einheitliches Bedienkonzept muss angestrebt werden.

4 | **Keine Änderungen der Onlinekommunikation der Emulatoren**
Für die Gültigkeit des FAT ist es von entscheidender Bedeutung, dass die Kommunikationsprotokolle der Emulatoren nicht beeinflusst oder verändert werden, da diese spezifisch auf die verwendeten Engineering-Werkzeuge und das zu emulierende System abgestimmt sind. Eine Vereinheitlichung der Kommunikation zur Laufzeit ist unzulässig.

2. GENERIERUNG EINER EMULATION

Wie in Abschnitt 1 erläutert, besteht die Grundidee des VEF darin, unterschiedliche Emulatoren in einer virtuellen Umgebung zu integrieren. Als Basis werden bestehende Virtualisierungstechnologien eingesetzt, da diese eine frei konfigurierbare Zusammenstellung des virtualisierten Hardware-Systems (wie Ethernet-Karte, Speicher) erlauben. Auf sich ändernde Hardware-Anforderungen kann flexibel reagiert werden, ohne neue reale Komponenten integrieren zu müssen. Die Umgebung ist flexibel skalierbar und weitgehend unabhängig von den Emulationsrechnern.

2.1 Parametrierung des VEF

Das VEF-Konzept sieht die automatische Generierung von virtuellen Maschinen (VM) und darin die automatische Instanziierung von Emulatoren vor. Die VM können im Anschluss an deren Generierung auf unterschiedliche real vorhandene PCs verteilt und ausgeführt werden. Ein im Rahmen des VEF entwickeltes Werkzeug übernimmt die Verteilung und die Steuerung der VM beziehungsweise der Emulator-Instanzen. Damit eine automatische Generierung der Emulation erfolgen kann, wird einmalig, wie in Bild 3 dargestellt, ein Muster einer VM aufgesetzt, auf welchem die unterschiedlichen Emulatoren installiert sind. Die Installation ist dabei genauso auszuführen wie auf einem realen PC.

Das vorbereitete Muster bietet den Vorteil, dass es beliebig vervielfältigt werden kann. Dadurch lässt sich die virtuelle Hardware automatisch verändern und auf die Bedürfnisse der jeweiligen Emulator-Instanz anpassen.

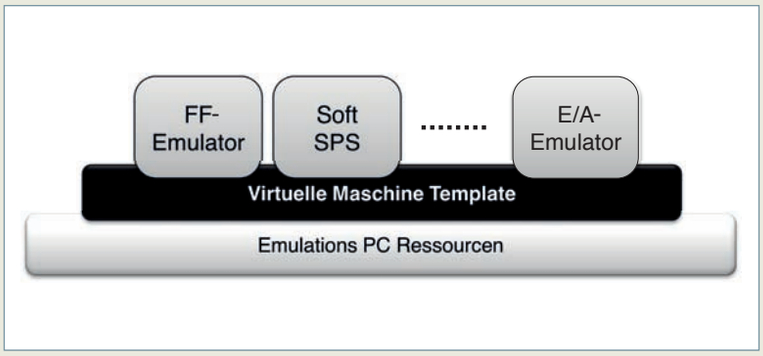

BILD 3: Template einer VM mit Emulatoren

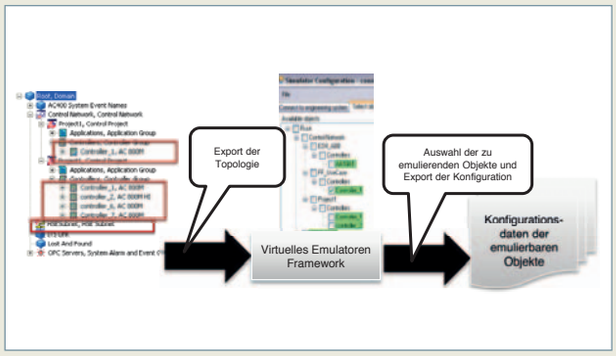

BILD 4: Export der AT-Topologie und Auswahl der zu emulierenden Objekte

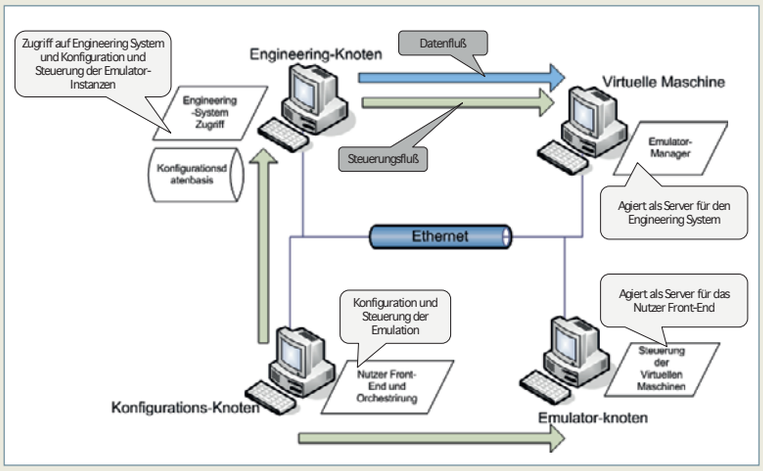

BILD 5: Software-Struktur des VEF

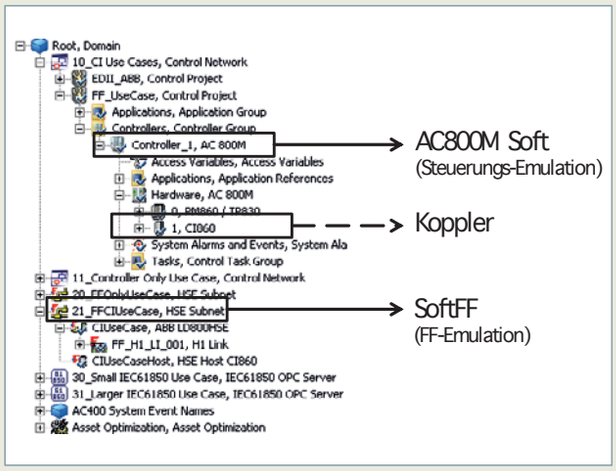

BILD 6: Topologie des AT-Systems

Eine Anpassung der externen Schnittstellen des realen PCs ist üblicherweise nicht nötig. Zusätzlich zur Installation auf dem Muster muss der Emulator die nachfolgend beschriebenen, von der spezifischen Konfiguration des Emulators unabhängigen Konfigurationsparameter bereitstellen.

1 | Anzahl der Ethernet-Schnittstellen: Nahezu jeder Emulator benötigt mindestens eine Ethernet-Schnittstelle. Mehrere Schnittstellen werden dann benötigt, wenn der Emulator mit einem Client-Server-Netzwerk des Leitsystems und einem FF-Netzwerk verbunden ist.
2 | Da Emulatoren instanziiert werden, welche lediglich in einer begrenzten Anzahl auf einem Rechner ausführbar sind, ist die Angabe dieser Begrenzung notwendig.
3 | Einige Emulatoren können mehrere Instanzen eines Subsystems ausführen. Diese Anzahl muss dem VEF bekannt sein.
4 | Hinzu kommt ein Parameter für den benötigten Arbeitsspeicher. Wie bei jeder MS-Windows-Applikation ist es für den Emulator wichtig, zu wissen, wie viel Arbeitsspeicher pro Instanz benötigt wird.
5 | Zur Identifikation eines emulierbaren Subsystems wird der zugehörige Objekttyp (zum Beispiel HSE-Subnetz für FF) benötigt. Auf Basis des Typs können die emulierbaren Systeme bestimmt werden.

Insbesondere bei der Emulation von Feldbussen werden die Anzahl der benötigten IP-Adressen pro Instanz sowie deren Adresswerte benötigt, da diese automatisch konfiguriert werden sollen. Letztere variieren mit der zu emulierenden Instanz und müssen dementsprechend aus den Konfigurationsdaten des Subsystems bezogen werden. Die Werte der erläuterten Konfigurationsparameter sind für jeden Emulator unterschiedlich und müssen daher bei der Integration des Emulators einmalig (mit Ausnahme der IP-Adresse) konfiguriert und bekannt gegeben werden.

2.2 Algorithmus zur Generierung der Emulation

Um die benötigten VM zu generieren und die Emulatoren darin zu instanziieren wurde ein mehrstufiger Algorithmus entwickelt. Jede Stufe benötigt dabei Informationen des jeweils vorangehenden Schrittes. So können die VM beispielsweise erst dann erzeugt werden, wenn deren benötigte Anzahl berechnet wurde.

Zunächst muss die Topologie des AT-Systems aus dem Engineering-System des Leitsystems exportiert werden. Die im Leitsystem modellierte Topologie enthält die AT-Komponenten, zum Beispiel die verwendeten Steuerungen. Da die emulierbaren Objekte aufgrund des Objekttyps der Emulatoren identifiziert werden, kann diese Information dem VEF bereits hier bekannt gegeben werden.

Während des FAT werden in der Regel Teilsysteme emuliert und getestet. Die Auswahl, welcher Teil getestet werden soll, kann der Nutzer mit dem in Bild 4 dargestellten Auswahldialog treffen. Dieser Auswahl entsprechend werden die Konfigurationsdateien für alle ausgewählten Objekte automatisch erzeugt. Die Dateien werden später benötigt, um die Emulator-Instanzen automatisch zu konfigurieren.

Auf Basis der ausgewählten Objekte lässt sich die Anzahl der benötigten VM unter Berücksichtigung folgender Regeln errechnen:

- Die maximale Anzahl ausführbarer Instanzen eines Emulators pro PC darf für keine der VM überschritten werden.
- Die VM haben abhängig von der verwendeten Virtualisierungsumgebung nur eine eingeschränkte Anzahl Ethernet-Schnittstellen zur Verfügung, die nicht überschritten werden darf.
- Die Anzahl gleichzeitig emulierbarer Objekte pro Instanz eines Emulators darf für keine Instanz überschritten werden.
- Der maximal verfügbare reale Arbeitsspeicher darf für keinen Emulationsrechner überschritten werden. Dies bedeutet, dass die Summe des für die VM eines PCs allokierten Speichers kleiner sein muss als der reale Speicher des PCs.

Da mit jeder zusätzlichen VM der Overhead für das Betriebssystem und die Virtualisierungssoftware steigt und für jede Betriebssystem-Instanz separate Lizenzen benötigt werden, wird empfohlen, stets die minimal notwendige Anzahl zu verwenden.

Nachdem die benötigte Anzahl an VM festgelegt und auf Basis des Musters erstellt wurde, muss die virtuelle Hardware entsprechend der Emulatoren, die in den VM ausgeführt werden sollen, konfiguriert werden. Die spezifischen VM enthalten anschließend die benötigte Hardware-Konfiguration sowie die emulatorspezifischen IP-Adressen. Hierdurch verhalten sie sich nach ihrem Start wie ein Emulations-Rechner, welcher auf die Emulatoren abgestimmt wurde. Im Anschluss an die Generierung der VM werden diese auf die Emulations-Rechner verteilt und dort gestartet. Die Konfiguration der Hardware ist damit beendet, worauf aufbauend die Dateien für die Konfiguration der generierten Emulator-Instanzen geladen werden.

2.3 Technische Umsetzung

Der beschriebene Algorithmus wird anhand einer Implementierung für das PLS-Engineering-Werkzeug System800xA erläutert. Als Emulatoren werden eine AC800M Soft-SPS sowie ein ABB SoftFF (FF-Emulator) in das VEF integriert.

Da System800xA eine verteilte Architektur aufweist und eine Realisierung des VEF mit Fokus auf komplexe ressourcenintensive Emulationen das Ziel ist, wird für das VEF ebenfalls eine verteilte Architektur gewählt. Als Basis für die Kommunikationsschnittstelle wird TCP/IP (beziehungsweise Ethernet) verwendet und auf der Windows Communication Foundation (WCF) [15] aufgebaut.

Grundsätzlich müssen für die Realisierung des VEF verschiedene PCs (im Folgenden als Knoten bezeichnet) betrachtet werden. Die Knoten stellen Funktionalitäten für die Konfiguration und Ausführung der VM und der Emulatoren bereit.

In System800xA werden dedizierte Engineering-Knoten verwendet, um das Engineering von der Laufzeit des PLS zu entkoppeln. Diese Knoten stellen die Informationen über die Topologie des AT-Systems bereit. Alle Informationen über die Engineering-Daten eines jeden Knotens werden zentral in einer Datenbank zur Verfügung gestellt. Es wird dementsprechend eine Software benötigt, welche Zugriff auf diese Datenbasis hat und mittels WCF mit dem Rest des VEF kommunizieren kann. In Bild 5 wird diese Software mit Engineering-System-Zugriff beschrieben.

Da aufwendige Emulationen üblicherweise hohe Anforderungen an die Hardware stellen, werden für diese ebenfalls dedizierte und separate Knoten eingesetzt. Diese werden in der technischen Umsetzung nicht in Form von Standard-PCs realisiert, sondern als private VMware ESXi Cloud [16]. Die Administration und Steuerung der virtuellen Maschinen innerhalb der privaten Cloud wird mit VMware vSphere erreicht [16]. vSphere bietet mit der VIX API (VIX Application Programming Interface, www.vmware.com]) und vSphere CLI (vSphere Command Line Interface) Schnittstellen für die Administration und Steuerung der VM. Diese sind jedoch nur lokal zugreifbar und müssen deshalb durch eine Software auf diesem Knoten netzwerkfähig gemacht werden. In Bild 5 wird diese als Steuerung der virtuellen Maschinen beschrieben.

Innerhalb der VM sollen die Emulator-Instanzen erzeugt, konfiguriert und gestartet werden, weshalb die in Bild 5 als Emulator-Manager bezeichnete Software notwendig ist. Zudem wird eine in Bild 5 als „Nutzer Front-End und Orchestrierung“ bezeichnete Nutzerschnittstelle benötigt, welche sich auf einem beliebigen, sich innerhalb des PLS-Netzwerks befindlichen Knoten befinden kann und die einzige Stelle darstellt, an der ein manueller Zugriff auf die Konfiguration der Emulation erfolgen muss. Die beschriebene Architektur besteht aus drei Servern mit unterschiedlicher Funktionalität und einem Client, der die Server mittels WCF steuert. Der Steuerfluss ist immer in Richtung der Server, siehe Bild 5.

Zusätzlich zum Steuerfluss wird ein Datenfluss zwischen dem Engineering-Knoten und den VM benötigt.

Die auf dem Engineering-Knoten erzeugten Konfigurationsdateien werden in den VM benötigt um die Emulatoren gemäß des in Abschnitt 2.1 beschriebenen Algorithmus zu konfigurieren. Um für einen späteren Einsatz des VEF in realen Anwendungen einen sicheren Datentransfer zwischen Engineering-PC und VM zu gewährleisten, wird empfohlen, den Datenaustausch über sichere Protokolle, wie zum Beispiel SSH (Secure Shell), durchzuführen.

2.4 Anwendungsbeispiel

Für die Ausführung des Algorithmus wird ein Beispiel verwendet, welches zwei verschiedene Emulatoren (Soft-FF [8] und Steuerungs-Emulation) benötigt. In Bild 6 sind die für dieses Beispiel verwendeten realen AT-Komponenten hervorgehoben und die für diese Anwendung benötigten Emulatoren dargestellt. Das AT-System besteht aus einer AC800M Steuerung sowie einem FF-Subnetz. Zur Kommunikation wird ein FF-Koppler verwendet.

Die hierfür im FAT vorgesehenen Testfälle sind beispielsweise:

- Test der Reglerparameter
- Test der Signalankopplungen
- Test des Alarm- und Event-Verhaltens
- Test der Ein- und Ausgangsfilter von FF
- Test der Verriegelungen
- Test der Grenzwerte

Für die Generierung der Emulation wird zunächst die Topologie des AT-Systems aus dem PLS-Engineering-Werkzeug exportiert. Hierfür greift das Konfigurationswerkzeug auf den Engineering-System-Knoten zu und benutzt dessen Server-Software, um einen Export zu initialisieren. Das Konfigurationswerkzeug stellt die erhaltene Struktur dar (siehe Bild 7).

Wie in Bild 7 dargestellt, wurden die Objekte „Controller_1“ (AC800M) und „21_FFCIUseCase“ (FF-Subnetz) bereits ausgewählt und basierend auf der Typinformation automatisch den Emulator-Typen zugeordnet.

Da für die technische Realisierung des Algorithmus eine EXSi Cloud als Laufzeitumgebung für die VM verwendet wurde, muss der Nutzer den Zugriff auf diese mittels Eingabe der URL einmalig konfigurieren. Anschließend wird die für das Anwendungsbeispiel benötigte VM aus dem Muster generiert, in die ESXi Cloud verschoben und dort konfiguriert.

Für das Beispiel werden zwei Ethernet-Schnittstellen, 64MB Arbeitsspeicher für den Soft Controller und 96MB Arbeitsspeicher für SoftFF benötigt. Die Schnittstellen und der Arbeitsspeicher werden mit Hilfe der VIX API automatisch hinzugefügt. Der Nutzer kann nun das Konfigurationswerkzeug anwenden, um die Emulation zu starten.

Im Vergleich zur herkömmlichen Vorbereitung und Konfiguration einer Emulation werden lediglich zwei Nutzerinteraktionen benötigt: 1) die Auswahl der zu emulierenden Komponenten und 2) die Konfiguration des Zugriffs auf die ESXi Cloud. Hinzu kommt, dass beide Instanzen – obwohl diese von unterschiedlichen Emulator-Typen gebildet wurden – auf die gleiche Art und Weise konfiguriert werden, wodurch keine weiteren Software-Werkzeuge verwendet werden müssen. Auch die virtuelle Hardware wird, basierend auf den Parametern der Emulatoren, automatisch angepasst.

2.5 Ressourcenbedarf und Anwendbarkeit des VEF

Im beschriebenen Beispiel wird lediglich eine VM benötigt. Jedoch erlaubt es bereits diese Konfiguration, Aussagen zu den benötigten realen sowie virtuellen IT-Ressourcen zu treffen. Um diese Angaben auf eine fundierte Basis zu stellen, wurden Benchmarks durchgeführt, mit Hilfe derer der Ressourcenbedarf während der Laufzeit für unterschiedliche Konfigurationen ermittelt werden kann.

Eine Konfiguration mit einer VM besteht nicht nur aus der VM, sondern aus der generierten VM und dem Muster. Das Muster wird für die Generierung nicht einfach kopiert; es werden Referenzen auf dieses gebildet, sodass die generierte VM ausschließlich den ergänzenden Speicherbedarf für die Konfigurationsdaten benötigt. Das Referenzieren auf das Muster stellt kein Problem für die Emulation dar, da diese im Arbeitsspeicher ausgeführt wird und während der Laufzeit keine Festplattenzugriffe benötigt.

Bild 8 zeigt den Ressourcenbedarf für die generierte VM. Das Muster ist mit 8GB erwartungsgemäß groß. Die generierte VM ist im Ursprungszustand jedoch nur 59MB groß. Erst nachdem weitere Zustände der VM eingefroren wurden, wächst der Speicherbedarf auf etwa 500MB an.

Außerdem verdeutlicht Bild 8, dass für die Emulation 416MB Arbeitsspeicher benötigt werden. Dies errechnet sich aus dem benötigten Speicher für die Emulator-Instanzen und den 256MB für das Betriebssystem. Als virtuelle Schnittstellen werden zwei virtuelle Ethernet-Karten gebraucht, je eine für die Soft-SPS und für Soft-FF.

Um die Rechenzeit zu ermitteln, wurde eine Testkonfiguration gemäß dem PassMark Performance Test 7.0 [www.passmark.com] auf einem Standard PC vorbereitet. Als Virtualisierungssoftware wurde VMware Workstation eingesetzt. Die Ergebnisse des Performance Test weisen lediglich einen kritischen Zustand auf: das Starten einer VM.

Die vielen Zugriffe auf die Festplatte, die während des Startens nötig sind, führen dazu, dass die Zugriffsgeschwindigkeit auf die Festplatte innerhalb der VM um etwa 30% einbricht. Auch die Zugriffsgeschwindigkeit auf den Arbeitsspeicher wird um zirka 20% langsamer. Es gilt daher, darauf zu achten, dass alle Startvorgänge von parallel laufenden VM vollständig abgeschlossen sind, bevor die Emulation gestartet wird.

3. DISKUSSION DES ANSATZES

3.1 Grenzen

Das VEF macht sich die Skalierbarkeit einer virtuellen IT-Infrastruktur zunutze. Hierdurch ergeben sich jedoch einige Ausschlusskriterien:

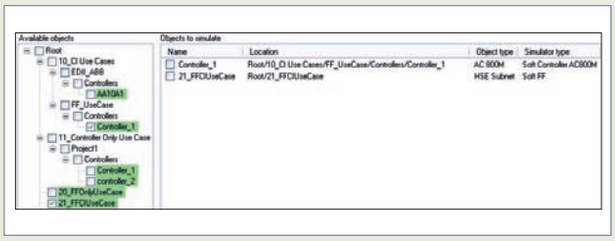

BILD 7: Screenshot: Auswahl der zu emulierenden Objekte

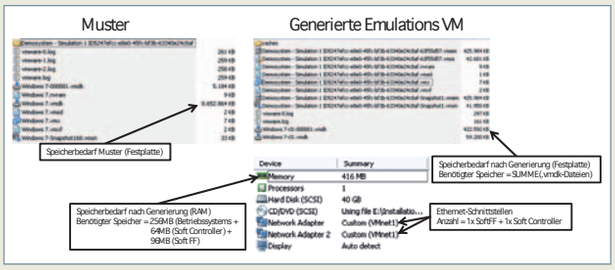

BILD 8: Benötigte Ressourcen für das Beispiel

- **Das VEF ist ungeeignet für hardwarebasierte Emulatoren.**
  Emulatoren, die als Basis eine Hardware-Plattform benutzen, können nicht in das VEF integriert werden.
- **Der Emulator muss auf einer Virtuellen Maschine lauffähig sein.**
  Generell müssen alle Emulatoren – auch softwarebasierte – die nicht in einer VM betrieben werden, können von einer Verwendung im VEF ausgeschlossen werden.
- **Ethernet als Kommunikationsbasis**
  Der Emulator sollte zur Laufzeit eine Ethernet-Kommunikation aufweisen. Emulatoren, die zum Beispiel über Profibus kommunizieren, benötigen spezielle Hardware, die nicht von der Virtualisierungsumgebung unterstützt wird.

Aktuelle Feldbusse basieren zunehmend auf Ethernet, weshalb davon auszugehen ist, dass Ethernet in Zukunft eine wichtige Rolle spielen wird [17] und hauptsächlich ältere Emulatoren betroffen sind. Emulatoren auf Basis von Hardware sind zwar nicht integrierbar, diese lassen sich jedoch über die konventionellen Kommunikationswege (beispielsweise Profibus PCI-Karte) im Verbund mit dem VEF verwenden.

3.2 Forschungsbedarf

Der Beitrag behandelt vor allem die Konfiguration der VM und der Emulatoren die darin ausgeführt werden. Aus wissenschaftlicher Sicht bleiben bezüglich der Generierung einer vollständigen Emulation folgende Fragestellungen offen:

- **Inwiefern wird das Laufzeitverhalten der Emulatoren durch die virtuelle Umgebung beeinflusst?**
  Um eine belastbare Emulation der Komponenten zu erhalten, müssen die Emulatoren ein mit der realen Komponente identisches Laufzeitverhalten aufweisen. Hierfür müssen Benchmarks entwickelt, angewandt und ausgewertet werden, welche die Emulatoren auf ihre Geschwindigkeit/Echtzeitverhalten untersuchen (zum Beispiel PLCopen Benchmark).
- **Wie können Emulationen geeignet gespeichert werden?**
  In der Regel wird eine Emulation für einen begrenzten Zeitraum benötigt und danach nicht weiter benutzt. Durch Persistieren könnte die Emulation zu einem späteren Zeitpunkt weiterverwendet werden (z.B. für Brown-Field Projekte).
- **Welche Schnittstellen zur Simulation werden benötigt?**
  Um eine vollständige virtuelle Inbetriebnahme zu ermöglichen, werden Schnittstellen für die Orchestrierung und den Austausch der Ein- und Ausgangssignale [3] zwischen Simulation und Emulation benötigt.

FAZIT

Neben den bekannten Vorteilen des Einsatzes von Virtualisierung – wie effizientere Hardware-Auslastung, geringere Betriebskosten, geringerer Stromverbrauch, geringerer administrativer Aufwand [16] im Vergleich zu konventionellen Rechnersystemen – bietet das VEF weitere Vorteile. Das VEF wurde im Hinblick auf eine effiziente Vorbereitung des FAT für AT-Komponenten und Subsysteme entwickelt. Der manuelle Aufwand zur Parametrierung, Konfiguration und Administration für die Emulator-Instanzen wird durch den Einsatz des VEF gering gehalten. Es werden lediglich zwei Nutzerinteraktionen benötigt: die Auswahl der zu emulierenden Objekte und zur Konfiguration des Zugriffs auf den ESXi-Server.

Durch den Einsatz des VEF erübrigt sich die Beschaffung, Montage und Parametrierung externer Rechnerschnittstellen. Die virtualisierte Hardware wird automatisch konfiguriert und parametriert, wodurch sich der Aufwand für die Bereitstellung der Emulation weiter verringert. Durch die Nutzung von Virtualisierung wird eine Lösung geschaffen, welche durch einen niedrigen Entwicklungsaufwand schnell zu einsetzbaren Werkzeugen führt. Auch die technische Realisierung zeigt, dass es möglich ist, eine Umsetzung für ein komplexes Leitsystem zu entwickeln. Das VEF ist damit eine skalierbare Umgebung, mit der Emulationen effizient für gewerkübergreifende, heterogene Leitsystemkonfigurationen erstellt und angewendet werden können.

MANUSKRIPTEINGANG
14.05.2012
Im Peer-Review-Verfahren begutachtet

REFERENZEN

[1] Rodies, H.-J.: Planungswerkzeuge aus Sicht des Anlagenbaus. atp – Automatisierungstechnische Praxis 44(1), S. 40-44, 2002

[2] Sato, H: The Recent Movement of Foundation Fieldbus Engineering. In: SICE Annual Conference, S. 1134-1137, 2003

[3] Barth, M.: Automatisch generierte Simulationsmodelle verfahrenstechnischer Anlagen für den Steuerungstest. VDI, 2011

[4] Greifeneder, J., Weber, P., Barth, M., Fay, A.: Generierung von Simulationsmodellen auf Basis von PLS-Engineering-Systemen. atp edition - Automatisierungstechnische Praxis 54(4), S. 34-41, 2012

[5] Maeda, T.: A testing environment. ABB's comprehensive suite of software testing and commissioning tools for substation automation systems. ABB Review Special Report IEC 61850, S. 29-32, 2010

[6] IEC61850: Communication networks and systems in substations, 2003

[7] Hoernicke, M. und Bauer, P.: Emulation dezentraler Steuerungslogik. Dynamisches Testen von Field-Control-Systemen. atp edition - Automatisierungstechnische Praxis 54(4), S. 42-49, 2012

[8] Hoernicke, M., Weemes, P., Hanking, H.: The fieldbus outside the field. Reducing commissioning effort by simulating FF with SoftFF. ABB Review 1/2012, S. 47-52, 2012.

[9] Günthner, W. und Ten Hompel, M.: Internet der Dinge in der Intralogistik. Springer, 2010

[10] VDI 3633: Simulation von Logistik-, Materialfluss und Produktionssystemen – Grundlagen, März 2000

[11] Wünsch, G.: Methoden für die virtuelle Inbetriebnahme automatisierter Produktionssysteme. Herbert Utz, 2008

[12] Barth, M., Fay, A., Wagner, F., Frey, G.: Effizienter Einsatz Simulations-basierter Tests in der Entwicklung automatisierungstechnischer Systeme. In: Tagungsband Automation 2010, S. 47-50. VDI, 2010

[13] IEC61131-3-Part 3: Programming Languages. Edition 2.0, 2003

[14] Brandao, D., da Cunha, M.J., Pinotti Jr., M.: Fieldbus Control System Project Support Tool based on Experimental Analysis and Modeling of Communication Bus. In: IEEE International Conference on Industrial Technology (ICIT'04), Vol. 2, S. 787-792, 2004

[15] Lowy, J.: Programming WCF Services. O'Reilly Media, 2008

[16] Runge, R., Sturm, C., Wisskirchen, S., Ebel, N., Groh, J., Höller, O., Mewes, C.: VMware Infrastructure 3 im Business-Umfeld. Addison-Wesley, 2009

[17] Morse, J.: Ethernet Steams Ahead in Asia Pacific. IMS Research, Januar 2012

AUTOREN

Dipl.-Ing. (FH) **MARIO HOERNICKE** (geb. 1984) ist Wissenschaftler am ABB Forschungszentrum in Ladenburg. Sein Forschungsschwerpunkt umfasst die Entwicklung neuer und innovativer Engineering-Konzepte im Bereich Emulation von Leitsystemfunktionen und Subsystemen sowie der Automation des Engineerings.

ABB AG Forschungszentrum
Wallstadter Str. 59, D-68526 Ladenburg
Tel. +49 (0) 6203 71 62 66
E-Mail: mario.hoernicke@de.abb.com

Dr.-Ing. **JÜRGEN GREIFENEDER** (geb. 1975) ist seit 2008 Wissenschaftler am ABB Forschungszentrum in Ladenburg. Er studierte Technische Kybernetik in Stuttgart und promovierte über formale Antwortzeitanalyse netzbasierter Automatisierungssysteme in Kaiserslautern. Seine wissenschaftlichen Schwerpunkte sind Systemmodellierung und effizientes Engineering.

ABB AG Forschungszentrum
Wallstadter Str. 59, D-68526 Ladenburg
Tel. +49 (0) 6203 71 62 22
E-Mail: juergen.greifeneder@de.abb.com

Dr.-Ing. **MIKE BARTH** (geb. 1981) ist Wissenschaftler am ABB Forschungszentrum in Ladenburg. Seine Forschungsschwerpunkte umfassen Methoden und Werkzeuge für ein effizientes Engineering von Automatisierungssystemen.

ABB AG Forschungszentrum
Wallstadter Str. 59, D-68526 Ladenburg
Tel. +49 (0) 6203 71 64 61
E-Mail: mike.barth@de.abb.com