



Werkzeugintegration und -verwaltung in heterogenen Computernetzwerken

Tool Integration and Management in Heterogeneous Computer Networks

Heinz-Josef Eikerling, Siemens Business Services

Wolfgang Müller, Universität Paderborn

Jan Wegner, Zuken GmbH

Mit der Verfügbarkeit von zunehmend komplexeren Computernetzwerken stellt sich auch immer mehr die Aufgabe der automatisierten Installation und Verwaltung von Anwendungssoftware im heterogenen Rechnernetz. Der vorliegende Beitrag stellt einen neuen Ansatz zur dynamischen Werkzeugverwaltung durch Kombination von SNMP, einem etablierten Standard zur Netzwerkadministration, und JINI-orientierten Konzepten am Beispiel der Integration und Verwaltung von Werkzeugen zum Leiterplattenentwurf vor. Abschließend diskutiert er alternative Ansätze zur Werkzeugverwaltung mit dem Schwerpunkt von existierenden XML-basierten Beschreibungssprachen.

Automation of tool installation and administration in heterogeneous computer networks becomes of increasing importance with the availability of complex computer networks. This article introduces a new approach for dynamic network tool management by combining a well established standard for network administration (SNMP) with concepts similar to JINI. The concepts are outlined by the example of the integration and management of design tools for Printed Circuit Boards (PCBs). A final discussion of alternative approaches focuses on existing XML-based description languages.

1 Einleitung

Die Werkzeugverwaltung und -integration [7] repräsentiert einen speziellen Aspekt bei der Erstellung von kollaborativen Entwurfsumgebungen [2; 10]. In diesem Umfeld stellen die Verwaltung und Integration von Ressourcen in komplexen – teilweise stark heterogenen – Computernetzwerken auch noch auf absehbare Zeit eine der großen Herausforderungen in der Systemadministration dar. Ressourcen in diesem Zusammenhang sind:

- physische Geräte und Hardware, wie Rechner oder Peripheriegeräte oder auch Werkzeugmaschinen,
- Software, wie zum Beispiel Unternehmensapplikationen, Entwurfsumgebungen oder Groupware,
- menschliche Ressourcen, also etwa Ingenieure, Programmierer oder Sachbearbeiter.

Um diese Ressourcen in einem komplexen, heterogenen Computernetzwerk zu verwalten und transparent verfügbar zu machen, sind weitergehende als die zur Zeit existierenden Ansätze erforderlich. Die An-

forderungen, die an derartige Konzepte und deren Beschreibungsmittel zu stellen sind, lassen sich wie folgt zusammenfassen:

- *Modularität/Erweiterbarkeit/Skalierbarkeit:* Um einen hohen Grad an Wiederverwendbarkeit sicher zu stellen, sollte die Integration einen modularen und ggf. hierarchischen Aufbau erlauben.
- *Verteilung und Portabilität:* Eine Werkzeugverwaltung muss dem Aspekt der dezentralen Verteilung der integrierten Ressourcen (z. B. Werkzeuge) ggf. auf verschiedenen Plattformen Rechnung tragen.
- *Internet-Fähigkeit:* Unter der Annahme, dass die Netzinfrastruktur zur Realisierung der Ressourcenverteilung durch das Internet bereitgestellt wird, sollte die Verwaltung an entsprechende Konzepte zur Datenbeschreibung und –übermittlung angelehnt sein.
- *Standardisierung:* Um den einfachen Austausch zwischen Integrationssystemen zu ermöglichen, sollten die Beschreibungsformate möglichst einer breit angelegten industriellen Unterstützung unterliegen.

Wir beschreiben in diesem Artikel einen neuen Ansatz zur dynamischen Ressourcenintegration und -verwaltung von Anwendungsprogrammen basierend auf SNMP (Simple Network Management Protocol), einem weit akzeptierten Standard zur Netzwerkadministration. Der Ansatz, der für den Anwendungsbereich der effizienten Entwurfsautomatisierung im Ingenieursbereich entwickelt wurde, stellt eine Kombination des JINI-orientierten *Discovery-Services* [4] mit einer SNMP-basierten Integration [8] und Registrierung von Werkzeugen dar. Zur Integration wird ein Anwendungsprogramm anhand weniger Charakteristika beschrieben, aufgrund derer der Discovery-Service die Vermittlung des Werkzeugs als Ressource im Netzwerk bereitstellt.

Die nachfolgenden Abschnitte strukturieren sich wie folgt. Zunächst gehen wir im Kapitel 2 kurz auf Grundlagen des *Simple Network Management Protocols* (SNMP) ein. Basierend hierauf stellen wir im Kapitel 3 unsere Konzepte zur dynamischen Werkzeugregistrierung und -vermittlung vor. Einige Details werden danach im Kapitel 4 anhand der Integration und Vermittlung von Werkzeugen zum Leiterplattenentwurf näher erläutert. Das abschließende Kapitel 5 verweist auf alternative und komplementäre Ansätze zur Entwurfsautomatisierung, welche im Bereich von XML-basierten Protokollen und in der Automatisierung von Arbeitsabläufen zu sehen sind. In diesem Bereich finden in jüngster Zeit immer häufiger sog. *Workflowmanagementsysteme* Anwendung, welche für wiederkehrende, gleichartige Arbeitsabläufe auf Basis einer Ablaufbeschreibung den automatischen Aufruf von Werkzeugen und den Datentransfer zwischen diesen realisieren.

2 Simple Network Management Protocol

Um Geräte in einem Netzwerk zu verwalten, hat sich seit 1989 das *Simple Network Management Protocol* (SNMP) als IETF¹-Standard speziell im Telekommunikationsbereich fest etabliert. Mittlerweile wird dieser Standard von einer Vielzahl von Soft- und Hardwareherstellern unterstützt. SNMP ermöglicht die Steuerung und Überwachung von Hardware- und Software-Komponenten in einem Netzwerk durch Austausch von Nachrichten. Hierzu liefert es zum einen Mittel zur Netzwerkintegration von Ressourcen mittels der Datenspezifikationsprache ASN.1² [5] und zum anderen einfache Mechanismen zum Austausch von Nachrichten [8].

Das Protokoll definiert auf Basis von UDP³-Diagrammen die Kommunikation zwischen mindestens

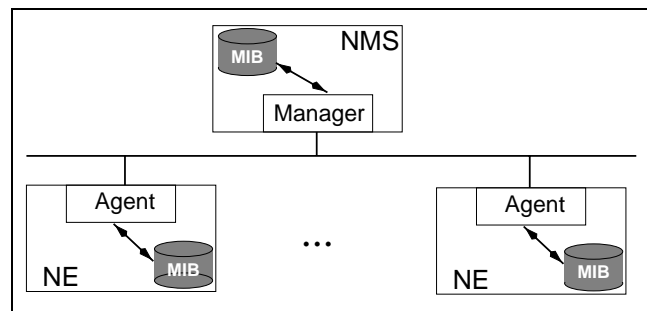


Bild 1: NMS und NE im SNMP-Netzwerk.

einer Netzwerkmanagementstation (NMS) und einer oder mehreren Netzwerkressourcen, sog. Netzwerkentitäten (NE). Bei den Entitäten kann es sich sowohl um einfache Geräte, wie z. B. Hubs oder Repeater handeln, aber auch um komplexere wie z. B. Switches, Gateways oder gar Arbeitsstationen. Im Detail kommuniziert der Manager einer Managementstation mit dem Agenten einer Entität durch den Austausch von Daten aus der jeweiligen Managementinformationsbasis (MIB) (siehe Bild 1).

SNMP-Nachrichten bestehen aus 2 Teilen. Der erste Teil beschreibt die SNMP-Version und den eindeutigen Bezeichner der sog. Community oder Party. Der zweite Teil beinhaltet die eigentlichen Daten (PDU – Protocol Data Unit) in Form einer Operation und der Objektinstanz. SNMP definiert fünf verschiedene Basistypen für Nachrichten:

1. **GET** – die NMS fordert eine Objektinstanz einer Netzwerkentität an
2. **GET-NEXT** – die NMS fordert die nächste Objektinstanz aus der Liste oder Tabelle von Netzwerkentitäten an
3. **GET-RESPONSE** – eine Netzwerkentität liefert als Antwort auf eine GET oder GET-NEXT eine Objektinstanz
4. **SET** – eine Netzwerkmanagementstation setzt Werte einer Netzwerkentität
5. **TRAP** – wird von einer Netzwerkentität als synchrone Nachricht an eine Netzwerkmanagementstation versendet

Das Format der Objektinstanz einer Nachricht wird mit Hilfe der Datenspezifikationsprache ASN.1 beschrieben. Die genaue Objektdefinition muss in der MIB (Management Information Base) der Netzwerkmanagementstation als auch der Netzwerkentität zur Interpretation der Objektinstanzen bekannt sein. In SNMP findet allerdings nur eine kleine Untermenge von ASN.1 Verwendung, die nur einfache Basistypen wie Boolean, Integer, Real und einfache zusammengesetzte Typen wie z. B. *Sequence* und *Set* umfasst. ASN.1-Spezifikationen für ein Gerät werden in der Regel von einem Gerätehersteller festgelegt und zur Verfügung gestellt.

Eine ASN.1-Definition beinhaltet die Definition eines Objektbezeichners, von Attributen (Datentypen,

¹ Internet Engineering Task Force

² Abstract Syntax Notation 1

³ User Datagram Protocol



Erläuterungen, Statusinformationen) und erlaubte Operationen. Der Objektbezeichner identifiziert ein SNMP-Objekt innerhalb des Netzwerks und innerhalb einer Managementinformationsbasis eindeutig.⁴ Er ist bezüglich einer abstrakten Baumstruktur hierarchisch aufgebaut und kann als Zeichenkettenfolge oder als Folge von ganzen Zahlen angegeben werden. So entspricht z. B. dem Objektbezeichner *iso.org.dod.internet.mgmt* die „Integer-Repräsentation 1.3.6.1.2“.

Der eigentliche Nachrichtenaustausch gehorcht einem einfachen Prinzip. Eine Managementstation pollt in der Regel in regelmäßigen Abständen die Netzwerkentitäten und fragt dabei vorher vereinbarte Werte ab. Die Entitäten liefern diese Werte, haben aber ansonsten keine Möglichkeit, die Managementstation zu manipulieren. Lediglich beim Auftreten unvorhergesehener Ereignisse ist es einer Entität erlaubt, eine Ausnahmenachricht – einen sog. TRAP – an die Managementstation zu senden. Dabei bleibt es aber der Station überlassen, ob und wie sie auf eine solche Benachrichtigung reagiert. Das übliche Vorgehen ist, dass die Station – z. B. ein PC – sich für den Status einer Entität – z. B. den eines Druckers und die Anzahl der gedruckten Seiten – interessiert. Dazu sendet die Managementstation eine GET-Nachricht an den Drucker und fragt nach dem Stand des Papierzählers. Erhält der SNMP-Agent des Druckers eine solche Anfrage, ermittelt er den Zählerstand und überträgt ihn mit Hilfe einer GET-RESPONSE-Nachricht. Um den Status des Druckers zu beeinflussen, kann sich die Station der SET-Nachricht bedienen.

3 Dynamische Werkzeugverwaltung mit SNMP

Das übliche Verfahren zur Werkzeuginstallation im lokalen Netzwerk besteht darin, dass der Systemverwalter Software auf einem zentralen Server im Netzwerk installiert. Neue Versionen von Anwendungen ersetzen entweder eine ältere Version oder werden auch zusätzlich zur Verfügung gestellt. So ergibt sich für den Benutzer eine sich ständig ändernde Arbeitsumgebung. Wegen der oft mangelnden Abwärts- und teilweise auch Aufwärtskompatibilitäten führt dies unter Umständen zu starken Herausforderungen in der täglichen Arbeit. Insbesondere in großen Netzwerken (Intranets) können deshalb Anwendungsprogramme als hochgradig dynamische Ressource angesehen werden, deren Nutzung flexibel und effizient verwaltet werden muss. Im Gegensatz zur Verwaltung von Hardwareressourcen stellt die Verwaltung und Vermittlung von Softwareressourcen ein wesent-

⁴ Objektbezeichner der Hauptstrukturen werden international von der IETF vergeben.

lich komplexeres Problem dar, da Software wie Hardware auch in verschiedenen Versionen, unter diversen Betriebssystemen und in unterschiedlicher Konfiguration (Standard, Professional, ...) vorliegen kann.

JINI [4] erreichte in jüngster Zeit im Bereich der Verwaltung und Vermittlung von Hardwareressourcen eine gewisse Popularität. Im Folgenden stellen wir vor, wie JINI-Basiskonzepte (Leasing, Discovery, LookUp) auf die Verwaltung von Werkzeugen übertragen und sinnvoll erweitert werden können. Werkzeuge sollen im Netzwerk zum einen durch ihren Namen, zum anderen durch Eigenschaften identifiziert und vermittelt werden können. Hierbei wird berücksichtigt, dass nicht nur ein Werkzeug vermittelt, sondern eine möglichst optimale Kombination von Werkzeug und Rechner identifiziert wird. Hierdurch wird es u. a. ermöglicht, Arbeitslasten effizienter im Netzwerk zu verteilen. Die Realisierung wird durch eine adäquate Kombination der Konzepte von JINI [4] mit dem SNMP-Standard [8] durchgeführt.

Im Einzelnen lässt sich die Werkzeugverwaltung konzeptionell in drei Teilaktivitäten unterteilen:

1. Registrierung
2. Vermittlung
3. Nutzung

Diese werden in den nachfolgenden Abschnitten näher erläutert.

3.1 Registrierung

Um im Netzwerk identifiziert werden zu können, werden Werkzeuge zur Registrierung beim *Tool Identification Server* (TIS) angemeldet (siehe Bild 2). Der TIS verwaltet eine globale, auf einer SNMP-MIB basierende Datenbasis. Dies bedeutet im Wesentlichen nichts anderes, als dass zur Nutzung von SNMP die Beschreibungen der verwalteten Werkzeuge als ASN.1-Spezifikation zu definieren sind. Zur Anmeldung eines Werkzeugs müssen zum einen die wichtigsten Charakteristika des Werkzeugs und zum anderen wichtige Grundfunktionalitäten bekannt sein. Die Bekanntmachung im Netzwerk und den Betrieb übernimmt für jedes installierte Werkzeug ein lokaler Prozess, der sog. *Tool Management Agent* (TMA). Dieser ist als Dämon konzipiert und wird bei der Installation eines Werkzeugs auf dem jeweiligen Rechner initialisiert. Die Grundfunktionalitäten des TMAs umfasst zunächst die Übermittlung der Werkzeugeigenschaften an den TIS bei der Registrierung. Des Weiteren müssen vom TMA periodische Anfragen vom TIS zur Verfügbarkeit beantwortet und bei der Auswahl des Werkzeugs der eigentliche Werkzeugaufwurf sowie die Übermittlung der Nutzdaten vorgenommen werden. Bei der Deinstallation des Werkzeugs übernimmt der TMA die Abmeldung beim TIS.

Parameter werden vom Tool Management Agenten in SNMP-Variablen verpackt und mittels eines SNMP-SET an den Tool Identification Server versandt. Dieser stellt das registrierte Werkzeug daraufhin in sein Repository, welches wiederum als Management Information Base (MIB) organisiert ist. Zur Registrierung und weiteren Verwaltung und Vermittlung werden Eigenschaften des einzelnen Werkzeugs sowie der Name des Rechners, auf dem das Werkzeug installiert ist, inklusive Betriebssystemdaten übermittelt. Im Einzelnen umfasst dies folgende Eigenschaften:

- Name,
- Versionsnummer,
- Installationspfad,
- Liste der möglichen Ein- und Ausgabeformate mit Versionsnummern,
- Liste der möglichen Parameter,
- Typ: edit, display, convert, ...,
- Interaktionsmodus: batch, shell, X11, win,
- Zeitintervall, in dem der TIS die Werkzeuginstallation überprüft.

Neben den konventionellen Eigenschaften eines Werkzeugs wie Name, Version, Installationspfad und Parametern ist es zur weiteren Charakterisierung des Werkzeugs sinnvoll, die Klasse und den erforderlichen Interaktionsmodus für einen möglichen Remote-Zugriff definiert zu haben. Die Angabe der Werkzeugklasse erleichtert das Suchen bzgl. allgemeiner Eigenschaften. Für unsere Zwecke schien die Typisierung in Display, Editor und Converter als ausreichend. Je nach Anwendung sind jedoch weitere Klassen vorzusehen. Der Interaktionsmodus gibt an, unter welcher Umgebung auf das Werkzeug zugegriffen werden kann. Wir unterscheiden hier den reinen Batchbetrieb (batch), den Betrieb mit zwingender ASCII-Ausgabe (shell), Ausgabe unter X11 (x11) und unter Windows-Oberflächen (win). Durch diese Angabe kann aufgrund des benutzten Betriebssystems entschieden werden, ob eine Werkzeugvermittlung sinnvoll ist. Arbeitet man z. B. unter Windows, ist ein Zugriff auf ein Werkzeug mit X11-Oberfläche nur sinnvoll, falls auf dem PC eine X-Terminalemulation zur Verfügung steht.

Ein wichtiger Punkt ist hier auch aus Performanzgründen die Spezifikation des sog. Pollingintervalls. Dies ist das Zeitintervall in Sekunden, in dem das Werkzeug (respektive der zugehörige TMA) vom Tool Identification Server auf Erreichbarkeit überprüft wird (vgl. Learning by JINI).

Zur effizienteren Vermittlung werden außerdem Eigenschaften des Hosts, auf dem das Werkzeug installiert ist, benötigt:

- IP-Adresse des Hosts
- Betriebssystemtyp und -version

Die folgende Tabelle gibt die Beschreibung der Eigenschaften exemplarisch anhand des Werkzeugs *Acroread* und des Windows-Editors *Notepad* wieder.

Tabelle 1: Eigenschaften verschiedener Werkzeuginstallationen.

NAME	Acroread	Acroread	Notepad
VERSION	3.0	3.0	4.0
TYPE	display	convert	edit
PATH	/usr/local/Acrobat3/bin	/usr/local/Acrobat3/bin	C:\winnt\notepad.exe
INPUT	pdf ?1.1?1.2?1.3?1.4	pdf ?1.1?1.2?1.3?1.4	txt
OUTPUT	ps ?Level1?Level2	ps ?Level1?Level2	txt
FLAGS	„default“%s, „help“-h	„default“-toPostScript%s, „help“-h	„default“%s
MODE	x11	batch	win
POLLING	3600 sec	3600 sec	3600 sec
HOST	131.234.80.61	131.234.81.61	131.234.81.66
OS_TYPE	sunos	sunos	winnt
OS_VERSION	5.6	5.6	4.0?3

Man sieht hier, dass *Acroread* als Anzeige-Display für PDF-Dateien und als Konverter von PDF nach PostScript klassifiziert werden kann. In beiden Fällen können unterschiedliche Ein- und Ausgabeformate akzeptiert werden. Diesen werden die unterstützten Versionsbezeichnungen mit Fragezeichen separiert hinzugefügt, wie z. B. PostScript?Level1?Level2. Optional können verschiedene Aufrufparameter (Flags) mit Bezeichnern angegeben werden. Es sollte jedoch darauf geachtet werden, dass zumindest der Standardaufruf (Default) und der Aufruf der Hilfefunktion (Help) spezifiziert ist.

3.2 Werkzeugvermittlung

SNMP kann nach der Installation der Werkzeuge zur

- Anfrage,
- Vermittlung und
- Nutzung

derselben eingesetzt werden, was Bild 2 schematisch darstellt. Ein Client wird im primitivsten Fall die einfache Suchanfrage eines Benutzers für ein bestimmtes Anwendungsprogramm realisieren, aber auch Anfragen aus Workflowmanagementsystemen sind hier denkbar. Im Folgenden wird die Anfrage und Vermittlung mittels des vorher eingeführten Tool Identification Servers (TIS) auf Basis von SNMP skizziert.

Das generelle Szenario beginnt mit der Formulierung einer Werkzeuganfrage. Innerhalb des hier definierten Ansatzes können konkrete Namen von Anwendungsprogrammen wie „Acroread“ evtl. mit Versionsnummer wie „3.0“ spezifiziert werden. Die hier

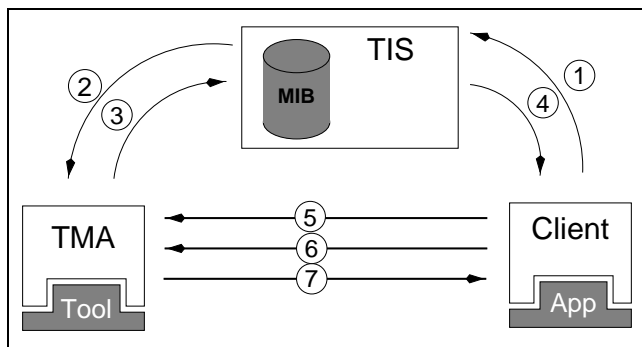


Bild 2: Werkzeuganfrage und -vermittlung.

vorgestellte Verwaltung erlaubt aber auch die alleinige Spezifikation von Eingabe- und Ausgabeformaten wie „Word6.0“, um das entsprechende Anwendungsprogramm vom TIS vermittelt zu bekommen. Des Weiteren ist es möglich, Eigenschaften zu kombinieren, sodass z. B. ein Editor mit Eingabeformat „PostScript“ angefordert werden kann. Die Anfrage eines Clients erfolgt mittels einer GET-Nachricht an den TIS (siehe Schritt 1 in Bild 2). Diese Nachricht enthält sowohl die eigentliche Anfrage als auch Eigenschaften des anfragenden Hosts wie z. B. Betriebssystem. Letztere werden zur nachfolgenden Werkzeugselektion benötigt. Der TIS konsultiert mit den Anfragedaten in Verbindung mit den Hosteigenschaften zunächst seine MIB-Datenbasis. Im Falle mehrerer Übereinstimmungen – z. B. kann ein Simulator auf mehreren Servern installiert sein – kann eine Auswahl bezüglich verschiedener Optimalitätskriterien erfolgen. So bringt der TIS aktuelle Konfigurations- und Laufzeitdaten wie Hauptspeichergroße und Prozessorauslastung mittels einer weiteren GET-Nachricht von den jeweiligen Hosts in Erfahrung (2). Die Hosts antworten mit GET-RESPONSE (3), worauf der TIS dann die endgültige Auswahl trifft. Die Referenz auf das vermittelte Werkzeug bzw. auf den Tool Management Agenten (TMA) des Werkzeugs wird nun mittels eines weiteren GET-RESPONSE vom TIS an den anfragenden Client weitergeleitet (4).

3.3 Nutzung

Nach der Werkzeugvermittlung erfolgt die Abwicklung der eigentlichen Nutzung der Werkzeuge. So fordert der Client den entsprechenden TMA mit einer GET-Nachricht auf, die Nutzdaten zu übertragen (5). Nach erfolgreicher Übertragung (6) startet der TMA das Werkzeug. Die Beendigung der Abarbeitung wird den Client per GET-RESPONSE inklusive etwaiger Fehlermeldungen mitgeteilt (7), der hierauf die Übertragung der Ausgabedaten initiiert, was den Anfrage-, Vermittlungs- und Nutzungszyklus beendet.

4 Anwendung

In diesem Kapitel wenden wir die vorher eingeführte Werkzeugvermittlung auf eine Entwurfsumgebung zum Erstellen von gedruckten Schaltungen auf Leiterplatten (engl.: PCBs – Printed Circuit Boards) an. Immer kürzer werdende Produktlebenszyklen – insbesondere im Bereich der Unterhaltungselektronik – erfordern einen hohen Grad an Automatisierung, welcher innerhalb komplexer Computernetzwerke mit der vorher vorgestellten Verwaltung noch erhöht werden kann.

4.1 Werkzeuge und Einsatz

Bild 3 skizziert die Arbeitsabläufe eines Leiterplattenentwurfsprozesses. Die eigentliche Komplexität ergibt sich hier nicht aus der Art der eingesetzten Werkzeuge, sondern aus der Anzahl der verfügbaren verschiedenen Versionen der Werkzeuge in Verbindung mit den verschiedenen Versionen der unterstützten – teils nicht kompatiblen – Ein- und Ausgabeformate.

Der Arbeitsablauf skizziert die Schritte vom Entwurf einer elektronischen Schaltung über das Layout der Leiterplatte bis hin zur Verifikation und Simulation

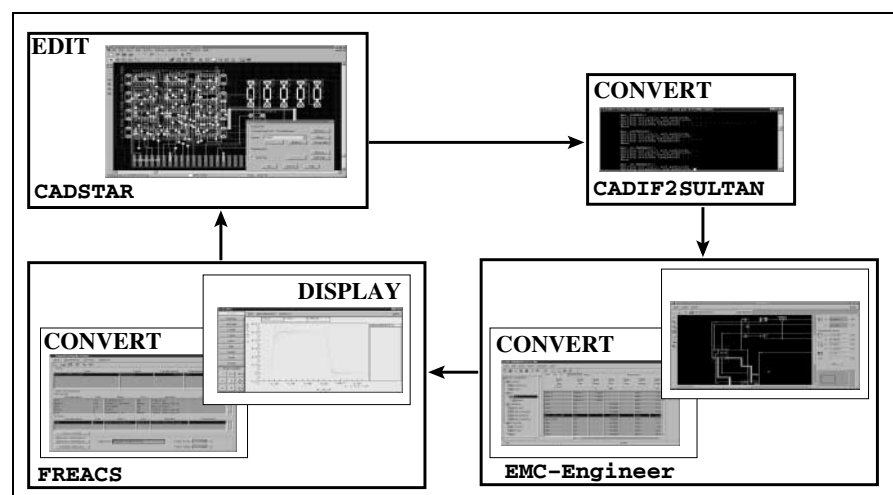


Bild 3: Vereinfachter Arbeitsablauf im Leiterplattenentwurf.

der elektrischen Eigenschaften des Leiterplattenlayouts. Wir bedienen uns hier exemplarisch der Software aus dem Hause *Zuken*, einem der führenden Hersteller auf diesem Gebiet. Es finden hier die Werkzeuge *CADSTAR-SI*, *EMC-Engineer* und die dort integrierten Simulationswerkzeuge *FREACS*, *Sigma* und *ComoRan* Anwendung. All diese Werkzeuge finden sich in der Softwaresuite *Hot-Stage* von *Zuken*.

Im Folgenden erläutern wir exemplarisch, wie sich ein Entwurfsablauf in unsere Werkzeugverwaltung integriert bzw. wie wir die einzelnen Arbeitsschritte in einen Arbeitsablauf unserer Werkzeugverwaltung integrieren. Als Grundlage sei in Bild 3 ein schematischer Entwurfsablauf gegeben.

Zur Erstellung des Entwurfs bedienen wir uns des *CADSTAR* Entwurf-Editors. Er stellt in der Produktpalette von *Zuken* ein sog. Low-Level-Entry-Produkt dar und ist als Standalone-Version rein PC-basiert. Mit Hilfe dieser Software lassen sich komplette Leiterplattenentwürfe erstellen. Daten werden sowohl in einem proprietären Format als auch im verbreiteten *CADIF*-Format abgespeichert. Der Konverter *CADIF2SULTAN* setzt die so erzeugten Daten in das *SULTAN*-Format als Eingabe für den *EMC-Engineer* um. Dieser stellt mit Hilfe von Benutzerinteraktion, z. B. manuelles Zuweisen von Komponenteneigenschaften, die Eingabedaten für die weiteren Simulationswerkzeuge bereit. Das Editieren der Entwurfsdaten für die anschließende Simulation findet auf einer UNIX-Workstation statt. Die Simulationsergebnisse werden hiernach von *ANARES* angezeigt.

4.2 Werkzeugintegration

Zur Integration der vorher spezifizierten Werkzeuge müssen diese nun in ihrer Installation definiert werden und die Beschreibung bei der Verwaltung (Tool Identification Server) angemeldet werden. Dies soll hier nur exemplarisch an der Beschreibung von *CADSTAR* vorgestellt werden. Wie schon vorher in Tabelle 1 eingeführt, bedarf es der genauen Angabe von Namen mit Versionsnummer, Werkzeugtyp, Installationspfad, Ein- und Ausgabe, Host-IP-Adresse, Interaktionsmodus sowie Betriebssystem mit Versionsnummer (und Service Pack):

```
cadstar;edit;c:\Programms\CADSTAR\cadstar.exe;
4.5.1;.pcb,.scm,.paf;.pcb,.scm,.paf?3,.paf?4;
win;196.22.22.22;winnt;4(3),
```

wobei in dieser Kurzform die einzelnen Felder mit Semikolon getrennt werden und die Versionsnummern den Formaten mit Fragezeichen angefügt sind. Man sieht hier, dass *CADIF* u. a. Dateien mit den Endungen „.pcb“, „.scm“ und „.paf“ erzeugen kann, wobei *CADIF*-Dateien (Endung .paf“) der Version 3

und 4 unterstützt werden. Ferner ist zu sehen, dass das Programm die grafische Oberfläche von Windows zur Interaktion benötigt (win).

4.3 Werkzeuganfragen

Auf Basis der vorher beschriebenen Integration und des Arbeitsablaufs wird im Folgenden eine kurze Sequenz mit vier Einzelanfragen an die Werkzeugvermittlung beschrieben:

1. CLASS=edit;INPUT=.pcb;OUTPUT=.paf?3; OS_TYPE=winnt
2. CLASS=convert;INPUT=.paf; OUTPUT=.sultan
3. NAME=emc-engineer;INPUT=.sultan; OUTPUT=.dia;MODE=X11; OS_TYPE=SunOS
4. CLASS=display;INPUT=.dia

Die Anfragen werden hier im ASCII-Format spezifiziert, wobei die vorangestellten Typnamen denen in der Tabelle 1 entsprechen.

In der ersten Anfrage wird ein Editor unter Windows NT für das *pcb*-Format gesucht, der als Ergebnis Version 3 des *paf*-Formats liefert. Die zweite Anfrage spezifiziert einen Konverter von *paf* nach *sultan*. Hiernach wird nach dem *EMC-Engineer* unter SunOS gesucht. Letztendlich soll die *dia*-Datei grafisch angezeigt werden. Das Beispiel zeigt, dass der Name des Werkzeugs nicht unbedingt bekannt sein muss. Es reicht zur Vermittlung die alleinige Angabe des Ein- und Ausgabeformats.

5 Alternative Ansätze

Zur Werkzeugintegration im Speziellen und zur Automatisierung von Arbeitsabläufen im Allgemeinen sind neben SNMP noch weitere Ansätze zur Integration bekannt, die hier noch kurz vorgestellt werden sollen. Es handelt sich hierbei zum einen um den *Tool Encapsulation Standard* (TES). Zum anderen gewinnen im Bereich der Ablaufautomatisierung SOAP und WSDL immer mehr an Bedeutung. Während TES teilweise in Konkurrenz zu dem in den letzten Kapiteln beschriebenen SNMP-basierten Ansatz steht, ist die Verwendung von WSDL als komplementär hierzu anzusehen.

5.1 Tool-Encapsulation Standard (TES)

Anfang der 90er Jahre wurde durch die *Common Framework Initiative* (CFI) ein Standard zur Werkzeugintegration definiert [2]. Dieser stellt einen Mechanismus zur Verfügung, der das Eingabe-/Ausgabeverhalten des zu integrierenden Werkzeugs beschreibt. Eine TES-Beschreibung für ein Werkzeug besteht aus:

- der Bezeichnung inklusive einer informellen Beschreibung des integrierten Werkzeugs,
- der Liste der Plattformen, die das Werkzeug unterstützen,
- dem Namen und dem jeweiligen Typ der Argumente wie z. B. String, Integer, Boolean,
- einer Klassifizierung der Argumente mit optionalen und erforderlichen Parametern,
- Präprozessoranweisungen zur Formatierung der Argumente, z. B. zur Konkatenation der formalen Parameter für den Aufruf eines Batch-Werkzeugs. Des Weiteren können Argumente durch Dialoge ermittelt werden. Die Auswertung der Interaktionen kann durch die Bereitstellung von sogenannten *Extension Codes* vorgenommen werden.

TES wurde als Standard in einer LISP-ähnliche Syntax definiert, wodurch sich die Implementierung und Interpretation von TES in LISP natürlich anbietet und eine höhere Portabilität und Flexibilität durch Interpretation erreicht wird. In diesem Sinne eignet sich TES auch gut zur Integration von komplexen Werkzeugen, speziell aus dem technischen Bereich, bei denen eine umfangreiche Parametrisierung vorzunehmen ist. Die eigentliche Anwendungsintegration – also z. B. die Verknüpfung eines Office-Werkzeugs und einer CAD-Anwendung – auf Ebene der Benutzungsschnittstelle wird hingegen nicht speziell unterstützt. Hier ergeben sich durch die Integration spezieller Programme zur Verwaltung der Interaktionen – z. B. Windows Scripting Host – insbesondere für Windows-Plattformen diverse Lösungsmöglichkeiten.

Auf ähnliche Art und Weise ist die Integration verschiedener Dienste denkbar. Hierunter sind Softwarekomponenten zu verstehen, die kontinuierlich unter einer definierten Schnittstelle und einem Protokoll einen Dienst – z. B. Datenbank oder Webserver – für andere Anwendungen zur Verfügung stellen.

Speziell im Hinblick auf die Integration von Webdiensten zeigt der TES-Ansatz in Bezug auf die Kompatibilität mit Web-Standards und der daraus resultierenden eingeschränkten Erweiterbarkeit und Mobilität der Integrationsbeschreibungen klare Defizite.

5.2 XML/WSDL

Ein wesentliches Kriterium für Beschreibungsformate ist die Internet-Fähigkeit des Integrationsansatzes. Die Annahme dabei ist, dass zukünftig eine Vielzahl von Diensten, also z. B. auch Zugriffe auf Entwurfswerkzeuge über das Internet, über Webschnittstellen bereitgestellt werden. Hieraus ergibt sich die Anforderung, die Kompatibilität des Integrationsformates mit den entsprechenden Standards zu gewährleisten.

In diesem Bereich ist eine zukünftige Dominanz XML-basierter Ansätze absehbar. Ein konkretes Integrationsformat wurde dabei über ein Firmenkonsortium mit der *Web Service Description Language* (WSDL) geschaffen. Hierbei handelt es sich um ein XML-Derivat, mit dem ein auf einem Webserver in Form von Skripten oder Anwendungscode implementiertes Objekt über einen Client durch Verwendung eines festgelegten Protokolls (Simple Object Access Protocol – SOAP) angesprochen werden kann. Beispielsweise lässt sich hierdurch ein Auftragsdienst zur Durchführung von Modellsimulationen durch einen *Application Service Provider* realisieren. Dabei kann davon ausgegangen werden, dass die Auftragsabwicklung mittels eines komplexen Arbeitsablaufs (Workflows) erfolgt, der seinerseits wiederum die Einbeziehung verschiedener Ressourcen (z. B. Webdienste) erfordert. Die Darstellung von Bild 4 skizziert die Prinzipien von WSDL anhand der Definition (Beschreibung der Nachrichtenformate) eines

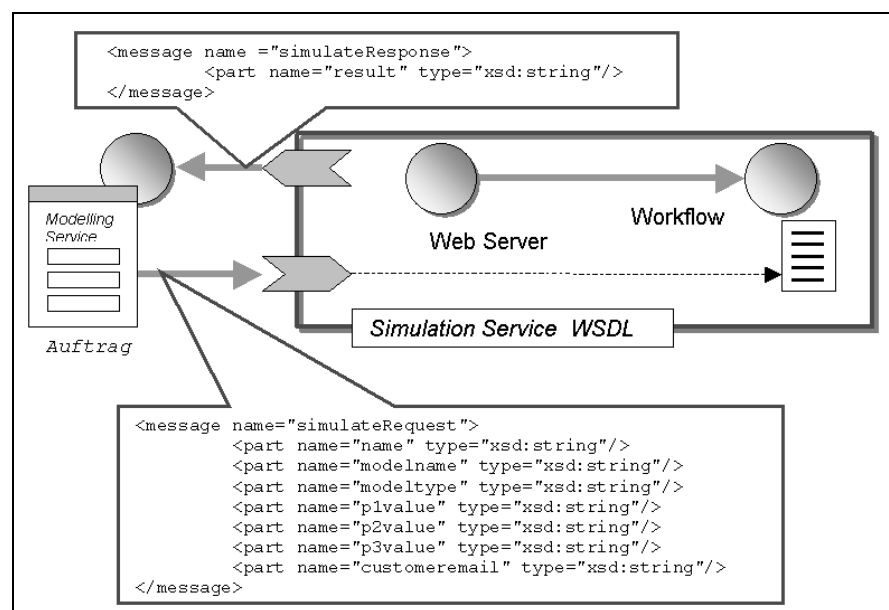


Bild 4: WSDL-Simulations-Webdienst.

Simulations-Webdienstes, wobei hier speziell auf die Attributtierung der Nachrichtenformate eingegangen wird. Eine Nachricht <message> wird dabei in verschiedene Komponenten (mit <part> gekennzeichnet) zerlegt, die die Strukturierung der übertragenen Daten widerspiegeln.

Innerhalb des WSDL-Formates werden ähnlich zu einer TES-Beschreibung neben dem Namen die Ein- und Ausgaben des Dienstes, sowie die Daten- und Nachrichtentypen definiert, über die ein Client diesen Dienst ansprechen kann. Genauer beinhaltet eine solche Definition die folgenden Elemente:

- *Types*: Datentyp-Definitionen unter Verwendung eines hierarchischen Typsystems
- *Message*: abstrakte, typisierte Definition der kommunizierten Daten
- *Operation*: abstrakte Beschreibung der unterstützten Aktion des Dienstes
- *Port Type*: eine abstrakte Menge der unterstützten Operationen
- *Binding*: ein konkretes Protokoll und Datenformat für einen bestimmten Port Type
- *Port*: ein einzelner Endpunkt, definiert als Kombination von Binding und Netzwerkadresse
- *Service*: eine Menge von zusammengehörigen Ports

WSDL-Beschreibungen lassen sich darüber hinaus über eine zentrale Registratur (UDDI – Universal Description, Discovery, and Integration) verfügbar machen, um so beispielsweise ein firmeninternes System zum Finden von internen Diensten zur Applikationsintegration aufzubauen.

Es bleibt zu erwähnen, dass von IBM unter dem Namen WSFL eine auf dem XML-Format basierende Integration von WSDL-Beschreibungen entwickelt wird [3].

5.3 Einsatz

Beide der vorgenannten Ansätze wurden im Workflowsystem von ASTAI(R) [6; 10] integriert. Basis des Systems ist ein CORBA-basierter Broadcast-Bus, der einen offenen Nachrichtendienst nach CFI ITC⁵-Standard realisiert (siehe Bild 5). Hierüber werden die einzelnen Komponenten des Systems miteinander gekoppelt.

Zentral ist die Arbeitsablauf- und -integrationslogik, die das Erstellen von Arbeitsabläufen über einen grafischen Editor sowie die Ausführung und Kontrolle von Arbeitsprozessen über eine sog. Workflow-Engine beinhaltet. Der Editor verwendet ein Repository für alle Teilkomponenten der Arbeitsabläufe, wie Aktivitätstypen, Werkzeugbeschreibungen nach TES-Standard, Daten-Klassen. Die Workflow-Engine ermöglicht den Zugriff auf die Laufzeitumgebung,

⁵ Inter-Tool Communication

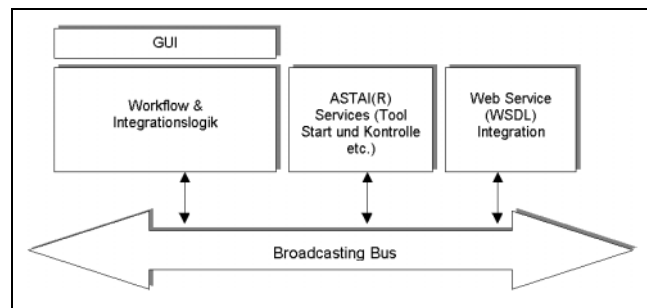


Bild 5: ASTAI(R) Broadcasting Bus.

die aus einem System kooperierender Dienste besteht. Ein wichtiger Dienst in diesem Zusammenhang ist der sog. Local Tool Launcher (LTL), der die in TES-Beschreibungen definierten Werkzeuge startet und dessen Ausführung kontrolliert. Die Laufzeitumgebung stellt sich dem Anwender als eine Art virtuelle Maschine dar, welche die Verteilung der Dienste auf Rechnerressourcen für den End-Anwender vollkommen transparent realisiert. So können je nach Plattform und sonstigen Anforderungen Werkzeugaufrufe über die LTLs den Rechnern zugeteilt werden.

Der Mechanismus zur Integration von Webdiensten in Arbeitsabläufe wurde dabei kürzlich zur Verfügung gestellt. Die Integration ist hierbei konzeptionell in beide Richtungen vorgesehen. Zum einen können ASTAI(R)-Arbeitsabläufe wie in dem oben dargestellten Beispiel als in andere Systeme integrierbare Ressourcen beschrieben werden, zum anderen ist es auch möglich, in WSDL beschriebene Webressourcen in ASTAI(R)-Arbeitsabläufe zu importieren.

6 Zusammenfassung

Dieser Artikel stellte einen neuen Ansatz zur Kombination des Simple Network Management Protocols (SNMP) mit den dynamischen Konzepten eines JINI-basierten Ressourcenmanagements vor. Als Anwendungsbeispiel diente die Integration von Werkzeugen für den Entwurf von Leiterplatten. Der vorgestellte Ansatz wurde erfolgreich in einer Implementierung mit dem *Java Dynamic Management Toolkit (JDMK)* [9] validiert. Im Zuge dieser Arbeiten konnte gezeigt werden, dass sich SNMP gut zur Übertragung kleinerer Werkzeuginformationen und -anfragen eignet. Bei einer Intranet-übergreifenden Werkzeugverwaltung sind jedoch die SNMP-inhärenten Sicherheitsdefizite zu berücksichtigen. Anwendungsspezifisch sind die vorgestellten Ansätze mit zusätzlichen Authentifizierungs- und Verschlüsselungsmechanismen zu kombinieren. Dies betrifft insbesondere den hier vorgestellten Bereich von Intranet-übergreifenden Entwurfsumgebungen, um die Entwurfsdaten gegen unerwünschtes Kopieren bzw. Abhören zu sichern.

Alternativ oder auch komplementär zum SNMP-basierten Ansatz bietet sich in einer Intranet-übergreifenden Lösung die Kombination mit dem hier vorgestellten WSDL-Ansatz an. Aber auch hier ist im Bereich industrieller Anwendungen aus den schon vorher genannten Gründen die Integration mit geeigneten Sicherheitsmechanismen zu realisieren.

Danksagungen

Die hier beschriebenen Arbeiten wurden teilweise durch Mittel des EU Projektes E-COLLEG (IST-1999-11746) gefördert.

Literatur

- [1] CFI: Tool Encapsulation Specification; Version 1.0.0. CAD Framework Initiative Inc., Austin, USA, 1992.
- [2] Lavana, H.; Brglez, F.; Reese, R.; Konduri, G.; Chandrakasan, A.: OpenDesign: An Open User-Configurable Project Environment for Collaborative Design and Execution on the Internet. IEEE Intl. Conference on Computer Design, 2000.
- [3] Leymann, F.: Web Services Flow Language (WSFL 1.0). IBM Technical Report, IBM Software Group, 2001.
- [4] Oaks, S.; Wong, H.: JINI in a Nutshell. O'Reilly Verlag, Köln, 2001.
- [5] Perkins, D.; McGinnis, E.: Understanding SNMP MIBs. Prentice Hall, USA, 1997.
- [6] Rammig, F.J.: Web-based System Design with Components Off The Shelf (COTS). Forum on Design Languages, Tübingen, Sept. 2000.
- [7] Schefstroem, D.; van den Broek, G.: Tool Integration. Wiley Series in Software Based Systems, John Wiley & Sons, 1993.
- [8] Stallings, W. B.: SNMP, SNMPv2, SNMPv3 and RMON 1 and 2. Addison Wesley Longman Inc., Reading, Massachusetts, 1999.
- [9] Sun Microsystems: Java Dynamic Management Kit 4.2 Tutorial. Sun Microsystems, Palo Alto, USA, Dez. 2000.

- [10] Thronicke, W.; Fox, W. et al.: From Tool Integration to Workflow Management – A Lean Integration Solution. In Proc. 2nd World Conference on Integrated Design and Process Technology, Austin, TX, Dez. 1996.



Dr. Heinz-Josef Eikerling studierte bis 1992 an der Universität Paderborn Informatik und wurde dort nach Tätigkeiten an den Universitäten Paderborn und Tübingen 1996 promoviert. Seit 1999 leitet er im C-LAB, einem gemeinsamen F&E-Institut der Universität Paderborn und des Bereichs SBS der Siemens AG, die Gruppe *Distributed Interactive Systems*. Seine Interessen umfassen Komponententechnologie und verteilte Systeme.

Adresse: Siemens Business Services/C-LAB, Fürstenallee 11, D-33102 Paderborn
E-Mail: heijo@c-lab.de



Dr. Wolfgang Müller studierte bis 1989 an der Universität Paderborn Informatik und wurde dort 1996 promoviert. Seit April 1997 leitet er im C-LAB die Gruppe *Visual Interactive Systems*. Seine Interessen konzentrieren sich zur Zeit auf methodische Ansätze und Beschreibungssprachen im Systementwurf.

Adresse: Universität Paderborn/C-LAB, Fürstenallee 11, D-33102 Paderborn
E-Mail: wolfgang@acm.org



Jan Wegner schließt in Kürze sein Studium der Informatik an der Universität Paderborn ab. Seit Juli 2000 arbeitet er bei der Zuken Deutschland GmbH, die sich mit der Implementierung und dem Vertrieb von Werkzeugen zum Leiterplattenentwurf befasst. Seine Interessengebiete umfassen Werkzeugintegration und die Administration von heterogenen Netzwerken.

Adresse: Zuken GmbH, Vattmannstr. 3, D-33100 Paderborn
E-Mail: Jan.Wegner@pad.zuken.de