

## Java<sup>ä</sup> und Eiblet<sup>ä</sup> Technologie im Internet

Robert Ott, Urs Nietlispach

JNet Systems AG

### Die www.eiblet.com Plattform

Was wäre eine neue Verbindungstechnologie ohne Standardisierung? Womöglich nur eine weitere Möglichkeit, Feldbus Systeme mit verschiedenen Übertragungsprotokollen mit einer standardisierten Aussenwelt wie dem Internet zu verbinden. Bei der Ausarbeitung der Eiblet Technologie wurde der Begriff Standardisierung bereits in der Konzeption berücksichtigt.

Seit Ende 1999 steht die Plattform www.eiblet.com der gesamten Welt offen zur Verfügung. Das integrierte Eiblet Diskussionsforum erlaubt den Zugriff und das Beitragen von Spezialisten aus aller Welt und ermöglichte, dass heute zwei API's (Application Programming Interfaces) für Java zur Verfügung stehen. Diesen zwei API's wurden die Namen 'Eiblet Bus API' und 'Eiblet Agent API' gegeben.

Das Eiblet Bus API definiert den busnahen Zugriff und beinhaltet bereits die spezifischen Klassen für das Bus System EIB. Erweiterungen des Eiblet Bus API's werden in naher Zukunft andere Bus Systeme wie Konnex (Convergence), BACnet, etc. beinhalten. Ziel der Definition des Eiblet Bus API's ist es, den busnahen Zugriff (meistens auf Bus Telegramm Ebene) für Java zu definieren. Es soll zudem sichergestellt werden, dass sämtliche Funktionalitäten der jeweiligen Bus Systeme unterstützen werden.

Das Eiblet Agent API definiert, im Gegensatz zum Eiblet Bus API, den busunabhängigen Zugriff. Dies erlaubt Systemen, die Eiblet basierend

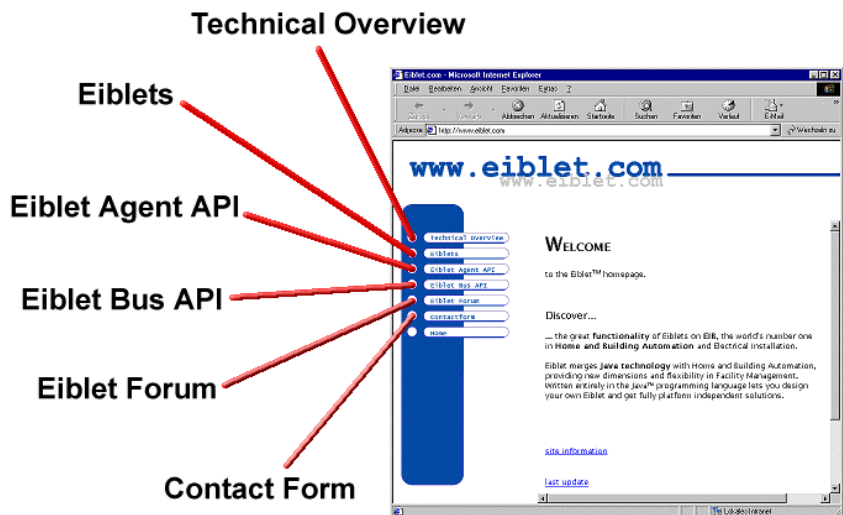


Bild 1. Die Hauptnavigationenpunkte der www.eiblet.com Plattform

aufgebaut sind, eine einfache Integration in netzwerkbasierende Umgebungen wie zum Beispiel dem Internet.

Das Eiblet Agent API erlaubt die Implementation von Applikationen, die unabhängig von den jeweilig eingesetzten Bus Systemen gehalten werden können. Trotz dieser Abstraktion muss auf Bus spezifische Funktionalitäten nicht verzichtet werden, da das Eiblet Agent API für einen solchen Bedarf neben den busunabhängigen Kommandos auch busspezifische Kommandos zur Verfügung stellt. Ein Beispiel eines Eiblet Agent API Kommandos ist das EibletCommandDimming, welches generelle Attribute wie 'increasing' oder 'decreasing' definiert. Davon abgeleitet ist das EIBCommandDimming, welches zusätzlich noch ein Attribut 'step-code', welches EIB spezifisch ist, zur Verfügung stellt. Natürlich definiert das Eiblet Agent API auch den Zugriff auf Datenpunkte eines Prozessabbildes von den entsprechenden Bus Systemen.

Neben den reinen Kommandos und dem Zugriff auf Datenpunkte definiert das Eiblet Agent API auch die

Schnittstelle für Logikbausteine, welche Eiblet's genannt werden. Ein Eiblet kann dynamisch in ein laufendes System hinzugeladen werden und Funktionalitäten wie Zeitschaltuhren, Licht- und Wohnszenen, periodische Speicherung von Messdaten und vieles mehr beinhalten.

Bild 1 zeigt die Hauptnavigationenpunkte der www.eiblet.com Plattform. Die Plattform informiert unter Eiblet Bus API und Eiblet Agent API über die erwähnten Programmierschnittstellen. Diese API's beinhalten auch Dokumentationen wie Tutorials und Links zu Implementationen der Eiblet Technologie. Weiter beinhaltet die Seite das Eiblet Forum, welches Entwicklern und Benutzern der Eiblet Technologie ein einfaches Mittel zum Informationsaustausch bietet.

### Die Eiblet Engine

Die Eiblet Engine ist ein Produkt der Firma JNet Systems AG und implementiert sämtliche API's, welche auf der www.eiblet.com Plattform beschrieben sind. Das Produkt stellt auch Schnittstellen wie WAP für Fernzugriffe via Mobiltelefon und

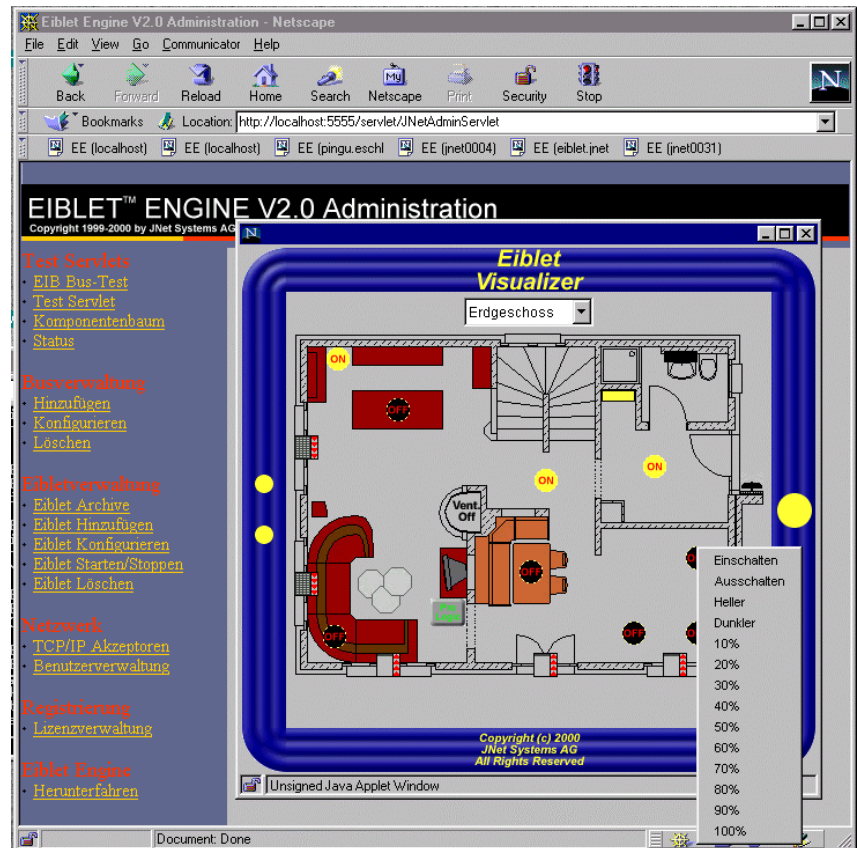
Robert Ott (34), Master of Science,  
Urs Nietlispach (31), Dipl. Ing. HTL,  
Firma JNet Systems AG, Postfach,  
9542 Münchwilen, Schweiz  
<http://www.jnetsystems.com/>

HTTP für Zugriffe durch einen Web-Browser zur Verfügung. Weiter können vorgefertigte Eiblets (Logikbausteine) dynamisch hinzugeladen, konfiguriert, aktiviert, deaktiviert und auch wieder gelöscht werden. Als Zusatzprodukt kann das Produkt Eiblet Visualizer zur graphischen Visualisierung und Fernsteuerung eingesetzt werden. Der Eiblet Visualizer kann einerseits als Applikation installiert und ausgeführt werden und andererseits auch via das Netzwerk als Java Applet in einen Browser geladen werden.

Die Eiblet Engine wurde vollständig in Java realisiert und wurde bereits auf verschiedenen Betriebssystemen wie Windows 9x/NT/2000, Linux und Embedded Linux getestet und eingesetzt. Als Run-Time Umgebung, die 7x24 Stunden aktiv sein sollte, wird als Grundlage für eine Eiblet Engine sehr oft Linux eingesetzt. Linux überzeugt vor allem im Embedded Bereich durch seine schlanke Implementierung und seine Stabilität. Die Visualisierung wird jedoch oft via Netzwerk auf einem normalen Windows PC oder einem iMac angezeigt und bedient.

Bild 2 zeigt die Administrationschnittstelle einer Eiblet Engine, welche durch einen Web Browser wie Netscape und Internet Explorer benutzt werden kann (siehe Hintergrundfenster). Durch diese Administrationschnittstelle können sämtliche Konfigurationsänderungen und Parametrierung vorgenommen werden. Es erlaubt Aktionen wie setzen und lesen von Werten auf dem Feldbus, Upload/Download von Eiblets, Parametrisierung von Eiblets, Berechtigungsvergabe und Netzwerk-konfiguration.

Die Eiblet Engine stellt das unterliegende Feldbus System als Prozessabbild zur Verfügung. Dadurch werden Leseabfragen von Statuswerten nicht durch das Bus System gebremst, sondern werden direkt von dem im Speicher gehaltenen Prozessabbild zur Verfügung gestellt. Schreibende Operationen werden natürlich direkt an das Bus System weitergegeben, welche in der Regel keinen wahrnehmbaren Verzögerungen unterworfen sind. Bild 2 zeigt auch ein Applet-Fenster, welches den oben erwähnten Eiblet



**Bild 2.** Die Administrationschnittstelle der Eiblet Engine und der Eiblet Visualizer

Visualizer zur Anzeige und Ansteuerung von Feldbusobjekten zeigt.

Bei der Implementation der Eiblet Engine als auch beim Eiblet Visualizer wurde die Datenbeschreibungssprache XML sehr gezielt eingesetzt. XML wird einerseits für jegliche Konfigurationen verwendet und erlaubt dadurch, dass die Eiblet Engine auch auf Kleinstgeräten lauffähig ist, wo schwergewichtige Programmelemente wie Datenbanken meist fehlen. Andererseits wird XML auch als Kommunikationsprotokoll zwischen verschiedenen Eiblet Engine 'Nodes' und zwischen einer Eiblet Engine und Produkten wie dem Eiblet Visualizer verwendet.

## JINI und OSGI

JINI und OSGI (Open Services Gateway Initiative) sind zwei komplementäre Technologien zu jener der Eiblets. JINI adressiert spontaneous Networking während sich OSGI als Standard Grundplattform für zukünftige Residential Gateways durchsetzt. Die Gemeinsamkeit von JINI, OSGI und der Eiblet Technologie liegt darin, dass alle drei Spezifikationen auf der Java Plattform aufsetzen.

JINI wurde im Frühjahr 1999 von Sun Microsystems Inc. als Spezifikation herausgegeben, um einen Standard zu definieren, wie sich Java basierende Geräte in einem Netzwerk auffinden können. JINI benutzt modernste Konzepte und Architekturen um diese Herausforderung zu lösen. Mit dieser Spezifikation wurde erreicht, dass sich bereits heute gewisse Arten von Geräten, die Java ausführen können, in einem Netzwerk auffinden können. Dies beschränkt sich jedoch heute noch weitgehend auf Geräte wie Drucker und Hellraumprojektoren. Mit dem JINI Projekt Surrogate wird im Moment ein Standard definiert, wie sich Geräte, die nicht fähig sind Java auszuführen, in einem JINI Netzwerk zur Verfügung stellen können. Dies wird in Zukunft erlauben, dass Feldbus-Komponenten wie EIB und LON via JINI auf dem Netzwerk zur Verfügung gestellt werden können. Damit wird das Auffinden von Standard Feldbus-Komponenten wie Lichtschalter, Heizungen und Temperaturmelder, die mit bereits bestehenden Protokollen kommunizieren, vereinfacht. Was dadurch noch nicht standardisiert ist, sind die jeweiligen Methoden (Funktionen), welche Ak-

tionen auf diesen Geräten ausführen sollen oder welche Status Informationen dieser Geräte zur Verfügung stellen. Genau dieses deckt die Eiblet Spezifikation ab und wird in Zukunft ermöglichen, das Lichter, Heizungen und andere Geräte einheitlich angesprochen werden können.

OSGI startete anfangs letzten Jahres mit einer Workgroup, bei der sich sämtliche namhaften Firmen zusammenschlossen, um einen Standard zu definieren, wie zukünftige Service Gateways (auch Residential Gateways genannt) aufgebaut sein sollen. Die Initiative konzentriert sich ausschliesslich auf die Softwarearchitektur, bei der natürlich Java als Grundlage angenommen wurde. OSGI definiert die Schnittstellen, die einzuhalten sind, wenn Services erstellt (programmiert) werden. Services die OSGI kompatibel sind, können prinzipiell in einen beliebigen OSGI Gateway hinzugeladen werden, sofern die für den Service nötigen Schnittstellen vorhanden sind.

Sobald OSGI Gateways auf dem Markt erhältlich sind, werden Services wie die Eiblet Engine in solche Gateways geladen und ausgeführt werden können. Natürlich kann die Eiblet Engine auch als Standalone Applikation auf einem Java fähigen Betriebssystem gestartet werden. Die Integration in Embedded Geräte wird jedoch in Zukunft durch OSGI Gateways erheblich vereinfacht werden.

## Online Demo mit WebCam

Bild 3 zeigt eine Online-Demonstration der Eiblet Technologie. Verschiedene Geräte können durch einen normalen Web Browser gesteuert werden, während der jeweilige Status des Raumes von einer WebCam (Kamera) aufgenommen wird. Das aufgenommene Bild der WebCam wird laufend an den steuernden Web Browser zurückgegeben und angezeigt.

Diese Demo-Anlage wurde mit Hilfe von verschiedenen Standard-Technologien realisiert. Für die elektrische Steuerung wurden normale EIB Komponenten verwendet, die auf dem Markt erhältlich sind. Als Web-Cam wurde eine AXIS Kamera ausgewählt, die bereits einen Web-Server beinhaltet. Dadurch kann die Web-Cam von einem beliebigen Betriebs-



Bild 3. Online Demo mit WebCam (<http://demo.jnetsystems.com/>)

system angesteuert und abgefragt werden.

Zur Steuerung der EIB Komponenten wird eine Eiblet Engine verwendet, die als Middleware und Gateway zwischen der Internet-Welt (TCP/IP) und der EIB-Welt (EIB Protokoll) agiert. Die Eiblet Engine beinhaltet auch einige Eiblet Logikbausteine, die Funktionalitäten wie Lichtszenen und andere Automationsfunktionen ermöglichen.

Die Applikation, welche die grafische Benutzerschnittstelle steuert, wurde ebenfalls mit Java realisiert. Auch hier wurde auf Standard API's nicht verzichtet. Das Programm benutzt einerseits das Eiblet Agent API zur Steuerung der Eiblet Engine und den dahinter liegenden EIB Komponenten und Eiblets. Andererseits wurde das HTML Browser Interface auf dem Standard der Java Servlets realisiert, welches in einer beliebigen Servlet-Umgebung (ServletExec, JRun, Tomcat, WebLogic, Apache JServ, etc.) ausgeführt werden kann.

Die Steuerung der Anlage erfolgt durch den Browser mit einfachen Mausklicks. Elemente wie Lichter, Rolläden, Szenen-Eiblets und die Pumpe der Wasserbeleuchtung können von der ganzen Welt aus von verschiedenen Benutzern gleichzeitig

angesteuert und angezeigt werden. Es wird sogar erlaubt, den Web Browser des PC's auf dem Tisch der Demoanlage zu steuern, indem man den Browser des PC's auf eine beliebige URL zeigen lässt. Dies erfolgt mit der Eingabe einer URL in das Textfeld unterhalb des WebCam Bildes und dem anschließenden Bestätigen mit dem Submit Knopf.

## Ausblick

Die Eiblet Technologie wird bereits in verschiedenen Projekten angewendet. Im Moment sind Produkte wie die Eiblet Engine bereits auf Embedded Buskomponenten als Prototypen lauffähig. In naher Zukunft wird die Integrationsmöglichkeit von Eiblet basierenden Systemen in Geräten wie Feldbus-Gateways und Residential Gateways (OSGI) völlig neue Massstäbe setzen, wie Gebäude und Eigenheime automatisiert werden!

## Web-Referenzen

**Demo:** [demo.jnetsystems.com](http://demo.jnetsystems.com)

**EIBA:** [www.eiba.com](http://www.eiba.com)

**Eiblet:** [www.eiblet.com](http://www.eiblet.com)

**Eiblet Engine:** [www.jnetsystems.com](http://www.jnetsystems.com)

**Java:** [java.sun.com](http://java.sun.com)

**JINI:** [www.jini.org](http://www.jini.org)

**OSGI:** [www.osgi.org](http://www.osgi.org)