Zukunftweisende Softwarelösungen für das Netzmanagement

Autor(en): **Lebsanft, Ernst**

Objekttyp: **Article**

Zeitschrift: **Comtec : Informations- und Telekommunikationstechnologie = information and telecommunication technology**

Band (Jahr): **74 (1996)**

Heft 7

PDF erstellt am: **05.08.2024**

Persistenter Link: https://doi.org/10.5169/seals-876777

CONSTRAINT SOLVING, KOMPONENTEN-SOFTWARE

ZUKUNFTWEISENDE SOFTWARELÖSUNGEN FÜR DAS NETZMANAGEMENT

Die technischen und kaufmännischen Anforderungen an das Management von Telekommunikationsnetzen nehmen ständig zu. Die Applikationsentwicklung mittels Softwarekomponenten eröffnet einen Weg, diesen Anforderungen zu begegnen. Eine spezielle, bereits durch marktgängige Komponenten unterstützte Technik ist das Constraint Solving. Es eignet sich besonders für Netzmanagement-typische Aufgaben wie Allokation von Ressourcen, Scheduling, Planning oder Konfiguration.

ERNST LEBSANFT, BINNINGEN BL

Das Management von Kommunikationsnetzwerken wird für ihre Anbieter immer kostenträchtiger. Moderne Telekommunikationsnetze werden zunehmend komplexer [1], getrieben durch die Deregulierung von Telekommunikationsdiensten in vielen Ländern und durch raffinierte Technologien wie SDH und ATM [10]. Die Komplexität des Managements solch heterogener Netzwerke nimmt entsprechend zu.

Man ist ferner bestrebt, den Servicegrad gegenüber den Kunden zu verbessern und gleichzeitig die Kosten zu reduzieren. Dazu werden komfortable rechnergestützte Managementsysteme benötigt, die es erlauben, schnell Kundenbedürfnisse zu realisieren, Überwachung und Funktionalitätserhaltung des Netzes zu gewährleisten, Administrations- und Wartungsaufwand zu reduzieren und das ganze Netz kosteneffizient zu betreiben [1].

Ausgangslage

Netzwerkmanagement ist eine auf Überwachung und Steuerung ausgerichtete verteilte Applikation nach ISO, bestehend aus Configuration, Fault, Performance, Accounting und Security Management. Es ist eine technische Aufgabe, in der das Zusammenwachsen von Informatik und Kommunikation sowie von Hardware und Software besonders deutlich demonstriert wird. Die Realisierung zukunftweisender Systeme in diesem Bereich erfordert Know-how in grafischen Benutzeroberflächen, Datenbanken, verteilten Systemen und Kommunikationsprotokollen. Bild 1 zeigt die grundlegenden Komponenten eines solchen Netzwerkmanagementsystems.

In diesem Beitrag wird zum einen mit dem Ansatz der Verwendung sogenannter *Softwarekomponenten* ein Weg zur Erreichung der erwähnten Ziele aufgezeigt. Zum anderen wird mit dem *Constraint Solving* eine inzwischen bewährte Technik vorgestellt, mit der eine Reihe von Problemen im Netzwerkmanagement inhaltlich besser als zuvor gelöst werden können.

Bisherige Lösungen für das Netzmanagement

Die bisherigen Lösungen sind entweder von Grund auf einzeln entwickelt worden oder basieren auf käuflichen Paketen. Telekommunikationssysteme sind nicht offen und nicht standardisiert. Sie enthalten oft ihre eigenen spezifischen Rechner, proprietäre Betriebssysteme und wenig flexible Anwendungssoftware.

In der Mitte der achtziger bis zu Beginn der neunziger Jahre waren überwiegend proprietäre, eigens entwickelte Systeme in Gebrauch, um dieser Heterogenität und der ihr innewohnenden Probleme Herr zu wer-

den. Ende der achtziger Jahre gab es eine Welle verschiedener herstellerspezifischer Plattformen und Protokolle, die bis heute angehalten hat. Standardisierungsgremien konzentrierten sich nur auf Kommunikationsprotokolle wie beispielsweise CMIP und SNMP. Kommunikation stellt aber nur einen Teil des Netzmanagements dar. Andere wichtige Aspekte sind beispielsweise die Benutzeroberfläche, die interne Systemkommunikation und die Datenspeicherung.

Zu dieser Zeit gab es kein vollständiges, leicht benutzbares Werkzeugpaket, das alle Aspekte der Realisierung eines Netzmanagementsystems abzudecken vermochte. Es gab keine grundlegende Architektur oder ein Rahmenwerk, das für Design und Implementierung unterstützend gewirkt hätte.

Man kann daher von einer Inadäquanz des gewählten Lösungsansatzes sprechen. Ein grosser Teil einer solchen *Softwareentwicklung «from scratch»* beschäftigte sich mit so allgemeinen Dingen wie Verteilung von Daten, GUIs und Interprozesskommunikation statt mit den applikationsspezifischen Aspekten für den Kern des Geschäfts. Die Anwendungen waren sehr spezifisch, oft sogar Hardware-spezifisch, und berücksichtigten die Netzwerkaspekte – aktuelle und zukünftige – nicht ausreichend.

Mit der Forderung nach schnellerer Entwicklung vollständiger Telekommunikationssysteme (Netzwerk–Hardware und Management–Software) gingen die Lieferanten zu *fertigen Paketen* über. Der Vorteil dieses Ansatzes liegt in der Beschleunigung der Entwicklung und in der Tatsache, dass das richtige Werkzeug an der richtigen Stelle verwendet wird. Werkzeuge (beispielsweise ein Konfigurationspaket) adressieren einen wohldefinierten Problemkreis (beispielsweise eine Konfigurationsmanagementapplikation) vollständig, und der Entwickler muss sich nicht um darunterliegende Dinge wie Architektur, Kommunikation usw. kümmern.

Probleme entstehen allerdings, wenn alle diese Teile zu einem integrierten Netzwerkmanagementsystem kombiniert werden sollen. Wenn die Pakete nicht von ein und demselben Lieferanten stammten, waren sie nicht interoperabel. Integration mit Paketen anderer Lieferanten war nicht vorgesehen. Die Folge waren (und sind noch) verschiedene GUIs, verschiedene Datenbanken und eine ineffiziente interapplikationskommunikation innerhalb eines vollständigen Systems. Die Systeme waren kompliziert in der Benutzung und hatten aufgrund ihrer komplexen Architektur einen entsprechenden Overhead.

Ein neuer Ansatz: Softwarekomponenten

Ein Weg aus dem Dilemma der Entwicklung proprietärer Systeme einerseits und der Benutzung und mühevollen Integration inkompatibler Pakete anderseits ist die Verwendung von Softwarekomponenten zur Optimierung von Make-or-buy-Entscheidungen.

Eine Softwarekomponente ist ein Softwarepaket, das eine wohldefinierte und begrenzte, aber typischerweise in *beliebigen* Applikationen vorkommende Aufgabe adressiert, und zwar genau diese Aufgabe, nicht mehr und nicht weniger. Eine Softwarekomponente ist also generisch. Man könnte sagen, dass Softwarekomponenten für Softwaresysteme das sind, was Chips für Computer.

Applikationsentwicklung geschieht dann weitgehend durch Zusammenbau, Konfiguration, Verfeinerung und Erweiterung von Komponenten. Beispiele von Komponenten in diesem Sinne sind Bibliotheken für die Erzeugung grafischer Benutzeroberflächen (GUI), generische Applikationsschnittstellen zu beliebigen RDBMS oder Klassenbibliotheken, die die Begriffswelt einer bestimmten Domäne modellieren.

Softwarekomponenten haben eine Programmierschnittstelle (API), durch die der Entwickler Zugriff auf die gesamte Funktionalität der Komponente hat. Dies ermöglicht eine leichte Integration in eine Applikation. Moderne Softwarekomponenten sind objektorientiert und können so vom Entwickler erweitert oder verfeinert werden. Beispielsweise könnte ein Allzweckikon erweitert oder vererbt werden, um als Knoten zu agieren. Heute erfolgreiche Softwarekomponenten sind oft in C++ geschrieben, was ihre Verwendbarkeit für jede Art von Applikation und Plattform erheblich erleichtert.

Charakeristika einer Softwareentwicklung mit Softwarekomponenten sind:

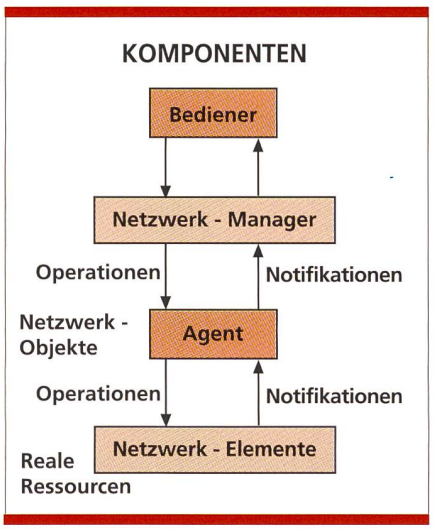

Bild 1. Komponenten eines Netzwerkmanagementsystems.

- ein hohes Mass an Wiederverwendbarkeit
- Portabilität
- hohe Qualität
- hohe Produktivität
- hohe Entwicklungsgeschwindigkeit

Auf dem Markt sind heute eine ganze Reihe bewährter generischer Softwarekomponenten verfügbar. Bekannte Hersteller sind beispielsweise Cosytec, I-Kinetics, ILOG, Interactive Objects oder TakeFive. Als Beispiele ziehen wir aufgrund eigener Erfahrungen die Komponentenpalette des Herstellers ILOG heran.

Views ist eine C++-Bibliothek für die Entwicklung beiliebiger grafischer Benutzeroberflächen mit interaktiven 2D-Grafikfähigkeiten. Anzeige und Editierung geographischer Information wie Karten und Lokationen, Handhabung von topologischen Daten wie Knoten und Verbindungen, von Anlageninformation wie Karten und Racks sowie von Managementinformation wie Alarmmeldungen, Performanzdaten usw. sind mit dieser Komponente möglich.

Server ist eine C++-Komponente für die Entwicklung reaktiver Client-Server-Applikationen. Server (in Kombination mit Views) erlaubt beispielsweise raffinierte, in Echtzeit synchronisierte GUIs mit mehrfachen Sichten derselben Daten. Server liefert eine grundlegende Architektur für Multi-

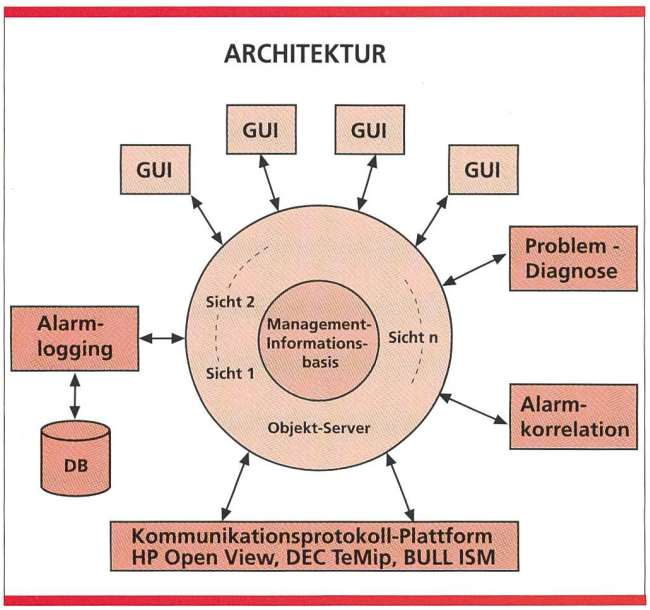

Bild 2. ILOG-Server-Architektur.

applikationssysteme. Einzelne Anwendungen kommunizieren nicht mehr direkt miteinander, sondern über *globale reaktive Objektserver,* wie Bild 2 zeigt. Diese stellen sicher, dass verschiedene Applikationen gemeinsame Informationen dynamisch und konsistent teilen (Bild 2). Für ein integriertes, alle fünf hier genannten Funktionsbereiche einschliessendes Netzwerkmanagement ist eine solche Komponente von besonderem Interesse.

Solver ist eine C++-Bibliothek für das Programmieren mit Randbedingungen (Constraint Satisfaction Programming, CSP), die weiter unten detaillierter beschrieben ist [2]. CSP eignet sich besonders für Ressourcenallokations- und Planungsaufgaben, also beispielsweise für die On-line Konfiguration von Netzwerken, die Konfiguration von Hardwareressourcen oder für das Routing. Der bei Solver gewählte deklarative Ansatz stellt sicher, dass das Modell zur Beschreibung einer Aufgabe von der Lösungssuche getrennt ist. Solver ermöglicht die Konfiguration von Ressourcen durch Deklaration dieser Ressourcen und der sie verbindenden Randbedingungen. Solver stellt die Einhaltung der Randbedingungen und der Konsistenz zu jedem Zeitpunkt sicher; dies ist nicht mehr Aufgabe des Entwicklers.

Schedule ist eine Zusatzkomponente zu Solver für die Entwicklung von Scheduling- und auch Ressourcenallokationslösungen. Schedule enthält vordefinierte Klassen zur Repräsentation von Aspekten des Scheduling, das heisst Schedule, Aktivität, Ressource, zeitliche Randbedingungen usw. Eine mögliche Anwendung ist beispielsweise ein System für das automatische Scheduling von Videokonferenzen zwischen mehreren Lokationen unter Berücksichtigung aller notwendigen Aspekte. Eine Fault-Management-Applikation auf Basis von Schedule könnte ein automatisches Rerouting um Problemzonen herum als Funktion enthalten. Bild 3 zeigt die Architektur von Solver und Schedule.

Während Views und Server für generische Systemfunktionen da sind, sind Solver und Schedule *spezifische, problemlösende* Softwarekomponenten. Um sie zweckmässig zu verwenden, muss eine Aufgabenstellung durch Objekte und Randbedingungen modelliert werden können. Dies ist nicht immer möglich, auch und gerade im Netzwerkmanagement. Fehlerdiagnose und Analyse von Datenströmen sind Beispiele sogenannter *schwach strukturierter Domänen,* das heisst, sie können nicht formal spezifiziert werden. Für solche Aufgaben haben sich *wissensbasierte Systeme,* in denen die Aufgabenstellung durch eine Regelbasis und einen auf ihr operierenden Inferenzmechanismus modelliert wird, bewährt. Typischerweise sind solche Systeme mit spezialisierten, oft wenig integrationsfreundlichen Werkzeugen realisiert worden [11, 12].

Rules ist eine weitere problemlösende C++-Bibliothek, die es erlaubt, eingebettete Regelbasen zu kreieren, und die über einen mächtigen Inferenzmechanismus, der direkt auf nativen C++-Objekten der Anwendung operiert, verfügt. Rules bietet sich daher für die Entwicklung fortgeschrittener Überwachungs-, Steuerungs- und Entscheidungsunterstützungssoftware an.

Ausser Views und Server gibt es weitere *Softwarekomponenten für die Entwicklung generischer Funktionen eines Systems.*

DB Link ist eine portable C++-Schnittstelle zu üblichen RDBMS, die eine multiple Konnektivität zu mehreren, auch unterschiedlichen RDBMS sicherstellt.

Broker ist eine Softwarekomponente für die Entwicklung verteilter Applikationen, in Kombination mit Server auch für Multioperator- und Multiscreen-Konfigurationen. Sie erlaubt dem Entwickler die transparente Manipulation entfernter Objekte, ohne dass er sich um die Kommunikationsschicht kümmern muss.

Bild 4 zeigt die Architektur eines mittels Softwarekomponenten aufgebauten Netzwerkmanagementsystems, das über regelbasierte Komponenten zur Filterung und Korrelierung von Ereignissen im Netzwerk (intelligente Netzwerkagenten) und zur Diagnose der Ereignisse (intelligente Netzwerkmanager), die die Agenten passiert haben, verfügt. Eine solche Architektur führt zu einer Reduktion der Anzahl von Alarmen und zu schnelleren, aussagefähigeren Diagnosen.

Ein zweiter Aspekt der gezeigten Architektur ist die Fähigkeit zur On-line-Rekonfiguration (beispielsweise ausgelöst durch die Rules-basierte Diagnose, ausgeführt durch die Solver/Schedule-basierte Applikation und koordiniert durch Server) während der Abwicklung eines Dienstes und zu einem entsprechenden Monitoring des Netzwerks.

Constraint Solving: eine «neue» Technik zur Lösung komplexer Probleme

Wie schon erwähnt, sind Allokation von Ressourcen, Scheduling, Planning oder Konfiguration sehr zentrale Aufgaben des Netzmanagements und der unterstützenden Systeme, typischer-

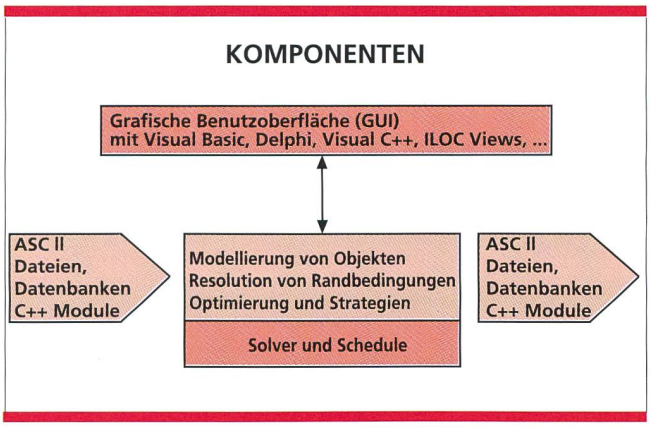

Bild 3. Architektur der Komponenten Solver und Schedule.

weise mit Optimierungs- und Echtzeitanforderungen. Probleme dieser Art sind kombinatorisch und daher von exponentieller rechnerischer Komplexität, was im Widerspruch zu den genannten Anforderungen steht. Traditionell sind solche Probleme durch Simulation, regelbasierte Ansätze und lineare Programmierung gelöst worden [2, 5].

Constraint Solving[1] [5–7] ist ein «neuer» Ansatz zur Lösung solcher Aufgaben, der so neu gar nicht ist, in unzähligen Applikationen weltweit den Nachweis der Praxistauglichkeit erbracht hat und heute in Form mehrerer Softwarekomponenten[2] kommerziell verfügbar ist.

1978 wurde der erste integrierte Constraint-Formalismus von J. L. Laurifre zur Lösung hochkombinatorischer Probleme vorgeschlagen. Mitte der achtziger Jahre wurde es zu einem «heissen» Forschungsgegenstand am European Computer Research Centre (ECRC) in München, aus dessen Prototyp verschiedene kommerzielle Werkzeuge hervorgegangen sind. Zur selben Zeit erweiterte A. Colmerauer seine in den frühen siebziger Jahren entwickelte Programmiersprache Prolog um das neue Paradigma. Auch aus diesen Forschungsarbeiten ging ein kommerzielles Produkt hervor. Constraint Solving ist also europäischen Ursprungs und hat jetzt grossen Erfolg in den USA, wie das rasche Wachstum beispielsweise von ILOG gerade in diesem Markt (40 % p.a.) zeigt.

Das Neue dieses Ansatzes liegt in der strikten Trennung der Repräsentation eines Problems vom Algorithmus zur Lösung des Problems. Ein Problem wird durch seine Unbekannten, das heisst seine Variablen, und eine Menge von Randbedingungen (Constraints), die durch diese Variablen erfüllt werden müssen, definiert. Der Constraint-Solving-Ansatz lässt sich also dann gut zur Lösung eines Problems verwenden, wenn gilt [5]:

- Das Problem lässt sich vollständig durch eine Liste von Variablen mit spezifizierten Randbedingungen beschreiben.
- Es ist mit einem iterativen und kooperativen Prozess, der die Randbedingungen verwendet, um unerlaubte Wertekombinationen für die Variablen zu finden, möglich, auf effiziente Weise zu Lösungen zu kommen.

Die Problemlösung besteht darin, Werte für die Variablen zu finden und dabei gleichzeitig die Randbedingungen einzuhalten, was bedeutet, dass technisch gesehen anspruchsvolle Suchprobleme[3] zu lösen sind.

Constraint Solving ist eine ziemlich mächtige und flexible Technik; sie erlaubt beispielsweise dem Applikationsbenutzer, Randbedingungen zu ändern, hinzuzufügen oder zu löschen, die sofort vom System propagiert werden und zu neuen Lösungen führen.

Bei Terminierungsaufgaben kann es vorkommen, dass es keine Lösung gibt, die sämtliche Randbedingungen erfüllt. Die CSP-Technik macht es möglich, mit dem «Aufweichen» an sich fester Randbedingungen (Constraint

[1] Dieser Begriff wird hier als Sammelbegriff für die verschiedenen Spielarten und damit auch Begriffe wie Programmieren mit Randbedingungen, Constraint Satisfaction Programming, Constraint Logic Programming, Constraint Programming usw. gebraucht.

[2] Beispielsweise Chip, Prolog II, Schedule, SNI Prolog, Solver.

[3] Die Bereitstellung hocheffizienter Suchalgorithmen zeichnet beispielsweise entsprechende Softwarekomponenten aus. Unserer Erfahrung nach dürfte es im allgemeinen kaum möglich sein, die Effizienz und Mächtigkeit angebotener Komponenten in diesem Bereich «auszustechen».

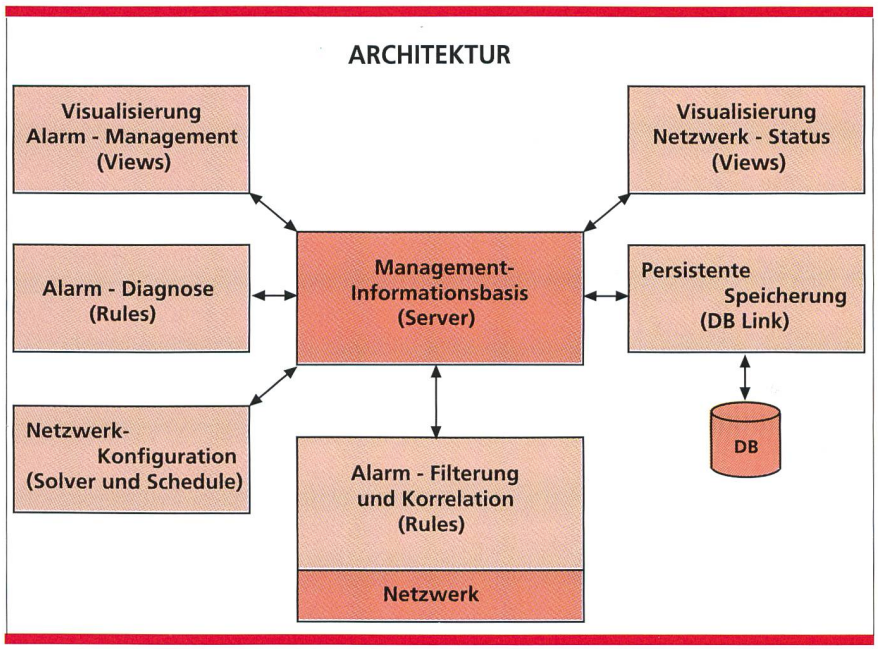

Bild 4. Architektur des Netzwerkmanagementsystems mit diversen Komponenten.

Literatur

[1] Benedikt Wälchli und Heinrich Blaser: Schritte zu einem dynamischen und flexiblen Management Network, ComTec 8/1995, S. 466–472

[2] Ernst Lebsanft: Programmieren mit Randbedingungen – eine neue Methode für komplexe Planungs- und Optimierungsprobleme, Bulletin des Schweizerischen Elektrotechnischen Vereins Nr. 25/1993, S. 11–14, 17. Dezember 1993

[3] Chee-Meng Low, You-tong Tan, Soo-Yong Choo, Sie-Hung Lau, Soo-Meng Tay: Autocell – an intelligent cellular mobile network management system, AI Mag., Vol. 16, No. 4 (Winter 1995), pp. 55–65

[4] Div. Dokumentationen der Fa. ILOG, Gentilly

[5] Peter Kibble: Fast solution through constraint solving, Objects in Europe, Vol. 2, No. 3 (May–June 1995), pp. 16–20

[6] Pascal van Hentenryck: Constraint satisfaction in logic programming, The MIT Press, Cambridge, MA, and London, 1989

[7] Vipin Kumar: Algorithms for constraint-satisfaction problems: A survey, AI Mag., Vol. 13, No. 1 (Spring 1992), pp. 32–44

[8] Joachim Hertzberg, Hans-Werner Güsgen, Angelika Voss, Manfred Fidelak, Hans Voss: Relaxing constraint networks to resolve inconsistencies. In: Joachim Hertzberg, Andreas Günter (Hrsg.) : Beiträge zum 2. Workshop Planen und Konfigurieren, Arbeitspapiere der GMD 310, St. Augustin, 1988, S. 91–101

[9] Jürgen Tenckhoff et al.: Objektorientierung zur effizienten Planung von Mobilfunknetzen, Objektspektrum 3/96, S. 16–25

[10] ComTec 8/95 mit mehreren Artikeln zum Thema ATM

[11] Ernst Lebsanft: Hybride Werkzeuge zur Entwicklung wissensbasierter Systeme, HARD AND SOFT 9/1989, VDI-Verlag, Düsseldorf

[12] Beat Liver, André Prim: Wissensbasierte Systeme im Netzwerkmanagement, Technische Mitteilungen der PTT, Bern, Januar 1992

Relaxation) zur Laufzeit mindestens Näherungslösungen zu erhalten [8]. Dessen Wert für die genannten Aufgabenstellungen ist offensichtlich.

Applikationsbeispiele

Die Telekommunikationsbranche hat zu Teilen das Potential dieser neuen Technologien erkannt, wie die nachfolgenden Beispiele zeigen mögen.

Singapore Telecom (ST) verwendet ein intelligentes Netzwerkmanagementsystem namens Autocell für ihr Mobilfunknetz [3]. Das System hat Effizienz, Kapazität und Servicegrad des Netzes verbessert und damit die Kundenzufriedenheit erhöht. Autocell ist eine mit dem Mobile Switching Center von ST verbundene verteilte Client-Server-Applikation mit folgender Kernfunktionalität:

- Speicherung, Aufbereitung und Präsentation von «Value-added»-Informationen über das Netz für Operateure, Techniker und Manager, das heisst beispielsweise Verkehrsprognosen
- automatische und dynamische Frequenzreallokation, zukunftsbezogene Optimierung
- Monitoring des Netzes
- Netzanalyse und -planung: Assistenz und Empfehlungen
- Performance-Überwachung, -Prognose und -Planung
- Konfiguration und Maintenance

Die Architektur von Autocell entspricht im wesentlichen der in Bild 4 gezeigten. Das System wurde in C++ und unter Verwendung von ILOG-Werkzeugen in zwei Jahren realisiert. Die Projektkosten wurden innerhalb des ersten Jahres wieder eingespielt.

France Telecom benutzt eine auf Solver beruhende Applikation zur Optimierung der Frequenzallokation für die Funktelefonie [4].

Sprint verwendet eine auf Rules beruhende Anwendung für Alarmbehandlung auf privaten Linien [4].

SIA, ein italienischer Netzwerkdienstleister, optimiert den Einsatz von Hardware- und Softwareressourcen des italienischen Interbankennetzes mit einem auf dem CSP-Ansatz beruhenden System [5].

Ebenfalls mittels Constraint Satisfaction Programming entwickelt *British Telecom* ein Planungssystem für den Einsatz von Ingenieuren und Technikern, mit dem die Anzahl zuzuordnender Aufträge maximiert und Überstunden sowie Reisekosten minimiert werden [5]. Eine ähnliche Anwendung verwendet die im englischen Telekommunikationsmarkt tätige Firma Energis [4].

CS-Telecom hat ein Überwachungs- und Sicherheitsmanagementsystem entwickelt, dessen Analysator für potentiell sicherheitsrelevante Ereignisse eine mit Rules entwickelte Wissensbasis enthält [4].

Beim deutschen Mobilfunkanbieter *DeTeMobil* wurden zunächst eigene Klassenbibliotheken, Frameworks und Softwarekomponenten geschaffen, um ein System für eine ziemlich umfassende Unterstützung der Aufgabe «Verwalten verschiedener Zustände des Sende- und Empfangsnetzes» zu realisieren [9]. Das Projekt wurde zu einem Zeitpunkt begonnen, als die bereits auf dem Markt verfügbaren geeigneten Softwarekomponenten im deutschsprachigen Raum noch kaum bekannt waren.

Ausblick

Die Verwendung dieser Technologien erlaubt erhebliche Steigerungen von Produktivität, Time to Market, Qualität, Flexibilität und Servicegrad gegenüber den Kunden, alles Beiträge zur Verbesserung der Wettbewerbsposition. Es wäre wünschenswert, wenn das Potential der beschriebenen, heute verfügbaren und bewährten Ansätze und Technologien auch hierzulande vermehrt erkannt und *jetzt* durch neue Applikationen ausgeschöpft würde.

Durch laufend neue und erweiterte Dienste für den Datentransfer, durch Text-, Bild-, Video- und Multimediakommunikation sowie mächtigere und flexiblere Netzwerke [10] gibt es immer mehr Anwendungsbereiche, in denen ein entsprechender Return on Investment zu erzielen ist.

Die Technologien wurden am Beispiel ihrer Verwendung im Netzwerkmanagement vorgestellt. Selbstverständlich sind sie nicht auf diesen Bereich beschränkt. Sowohl Applikationsentwicklung mit Softwarekomponenten allgemein als auch die Constraint-Sol-

SUMMARY

Constraint solving

The technical and commercial requirements to be satisfied by the telecommunications network management are steadily growing. Applications development by means of software components is a viable means for coping with these requirements. A special technology, already supported by marketable components, is constraint solving. It is specifically suited to network management tasks such as resource allocation, scheduling, planning or configuration. This technology considerably boosts the productivity, time to market, quality, flexibility and degree of service available to the customers and therefore enhances the competitiveness. It would be advantageous if the potential of the currently available and proven solution approaches and technologies described in this report were better understood also in our country and exploited now through new applications.

ving-Technik sind Querschnitt-Technologien, die sich sehr vielfältig verwenden lassen, wie nicht nur unsere Erfahrung in vielen verschiedenen Branchen und Anwendungsgebieten zeigt. [9.4, 11]

Ernst Lebsanft, Dipl.-Phys., Dr. rer. nat. (Univ. Bonn), ist Gründer und Geschäftsführer der SYNLOGIC AG, einer auf die Entwicklung kundenspezifischer Softwarelösungen spezialisierten Software- und Beratungsfirma. Der praktischen Umsetzung fortgeschrittener Softwaretechniken kommt im Geschäft von SYNLOGIC eine besondere Bedeutung zu.