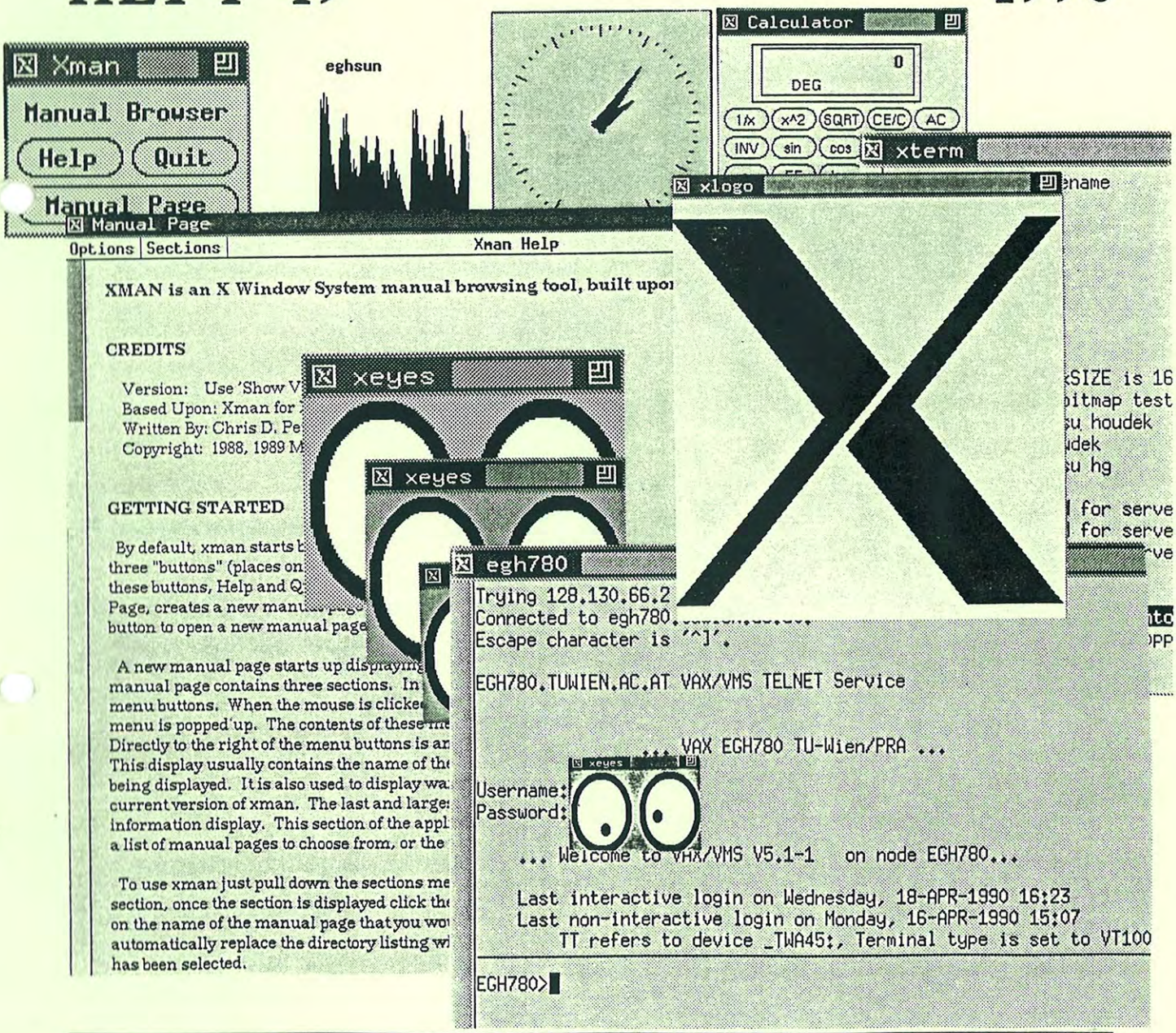
FEEDBACK

HEFT 19 1990



TU-PRA

Herausgegeben von der Abt. Prozeßrechenanlage des EDV–Zentrums der Technischen Universität Wien, A–1040 Wien, Gußhausstraße 25

Editorial

Mehr als fünfzehn Jahre lang hat die Zeitschrift "Feedback" einen Überblick über die verschiedenen Arbeitsgebiete der Abt. Prozeßrechenanlage gegeben. Die vorliegende Ausgabe setzt diese Tradition noch einmal fort, wird aber gleichzeitig die letzte in dieser Form erscheinende Nummer sein.

Die Anforderungen der Benutzer an ein universitäres EDV-Zentrum haben sich in den vergangenen Jahren, bedingt durch die Dezentralisierung der Rechenleistung, stark gewandelt. Aus diesem Grund wird derzeit versucht, im Rahmen einer Reorganisation des EDV-Zentrums die bisherigen drei Abteilungen (Digitalrechenanlage, Hybridrechenanlage, Prozeßrechenanlage) in ein einheitliches Rechenzentrum überzuführen, um den geänderten Anforderungen der Benutzer sowie den neuen technologischen Entwicklungen Rechnung zu tragen.

Die Beiträge in dieser letzten Ausgabe des "Feedback" behandeln wie immer wesentliche Neuigkeiten bzw. Schwerpunkte aus der Arbeit der Abt. Prozeßrechenanlage. Das Fachgebiet CAD und Graphik, das in den letzten Jahren intensiv betreut wurde, ist in diesem Heft auch wieder stark vertreten.

Benutzer der Calma-Anlage, die mehr als zehn Jahre lang für die Leiterplatten-Entwicklung zur Verfügung stand, erhalten einen chronologischen Rückblick von den ersten Anfängen mit dem NOVA/GDSI-Rechner bis zum Nachfolgesystem von SUN.

Der Beitrag über EUCLID-IS informiert über sämtliche Möglichkeiten, die dieses 3D-Volumenmodell, das erstmals mit allen Modulen in der aktuellen Version zur Verfügung steht, im Bereich der mechanischen Konstruktion bietet.

X-Window, die graphikfähige Benutzeroberfläche der Workstation-Welt, wird kritisch unter die Lupe genommen, und es sei in diesem Zusammenhang auch auf den Beitrag über die Leihstellung einer Tektronix-RISC-Workstation hingewiesen.

Vervollständigt wird die letzte Ausgabe des "Feedback" noch durch aktuelle Berichte über Datennetze, Software-Engineering und Desktop Publishing, wobei schon jetzt gesagt werden darf, daß die Information der Benutzer über die Aktivitäten eines einheitlichen, reorganisierten EDV-Zentrums der Technischen Universität auch in Zukunft in Form neuer Publikationen fortgesetzt werden wird.

G.W.

Inhaltsverzeichnis

Offenlegung gemäß §25 Mediengesetz

Grundlegende Richtung

Information der Benutzer der Rechenanlagen an der Abt. Prozeßrechenanlage des EDV-Zentrums der TU-Wien

Inhaber, Herausgeber

EDV-Zentrum der Technischen Universität Wien,
Abt. Prozeßrechenanlage,
A-1040 Wien, Gußhausstraße 25-29

Hersteller

Druckerei der ÖH an der TU-Wien,A-1040 Wien, Wiedner Hauptstraße 8-10

Redaktion/Layout

Dr. G. Wehrberger, Dipl. Ing. G. Gollmann

Diese Ausgabe enthält eine Werbeeinschaltung der Firma Tektronix Ges.m.b.H.

"X-Windows": Konzepte, Erfahrungen

Walter Selos

Bezugnehmend auf die im vergangenen Herbst an der **TU** stattgefundene "X-Window"-Vorführung möchte ich zum Thema "X-Window" einige Gedanken festhalten.

Der Name ist heute in aller Munde, speziell im Zusammenhang mit Unix-Workstations (aber auch VMS-Stations blieben nicht verschont), die grundlegenden Konzepte scheinen aber weniger bekannt zu sein. Der folgende Artikel soll in erster Linie eine Planungshilfe und Übersicht darstellen, und soll mithelfen, die sich bietenden Möglichkeiten als wirkungsvolles Werkzeug einzusetzen, und nicht als ressourcenfressendes Ungeheuer, das man hat, weil's halt jeder hat und modern ist, ohne sich völlig im klaren zu sein, wozu man es überhaupt braucht.

Worum geht es eigentlich:

"X" ist ein spezielles Kommunikationsprotokoll zwischen zwei Prozessen, die nicht unbedingt am selben Rechner laufen müssen. Dieses Protokoll ist auf den Austausch von jenen Daten spezialisiert, welche zwischen irgendeiner Applikationssoftware und deren (graphikfähiger) Benutzeroberfläche anfallen. Man spricht in diesem Falle von der Applikationssoftware als "client", und von der Software, welche die Benutzeroberfläche (Bildschirm, Maus,...) bedient als "server". Diese beiden Einheiten können, aber müssen nicht unbedingt am selben Rechner liegen. Das Transportprotokoll über das Netzwerk ist dabei nicht festgelegt, in den meisten Fällen wird TCP/IP verwendet. Betrachtet man nun das bisher Gesagte, können wir schon die grundsätzlichen Vor- und Nachteile erahnen.

Zu den wesentlichsten Vorteilen gehören:

- ein genormter Datenaustausch zwischen Applikation und der Mensch-Maschine-Schnittstelle, daraus folgt:
- eine leichtere Portierbarkeit von Software zwischen verschiedenen Workstations.
- Die Netzwerkfähigkeit dieses Datenaustausches und somit
- eine bessere Verteilung und Ausnützung der Ressourcen mehrerer vernetzter Workstations.
- durch das Windowkonzept: die Möglichkeit einer gleichzeitigen Bedienung mehrerer Prozesse am selben Display.
- Menüsteuerungen können durch Einträge in eine Konfigurationsdatei des Windowmanagers leicht und schnell selbst hergestellt und auch verändert werden - eine Möglichkeit, die Entwicklungskosten einspart.

Als Nachteile ergeben sich:

- eine große Bindung von Ressourcen: Die Serversoftware (Server, Windowmanager) benötigt relativ viel Arbeitsspeicher, ebenso werden die Applikationen nicht nur durch das Dazubinden von speziellen "Library-modulen", sondern auch durch die Notwendigkeit von mehr "sourcecode" größer. Es ist auch evident, daß der Betrieb dieser relativ komplexen Software auch einiges an CPU-Zeit kostet. Die Anzahl der Prozesse steigt ziemlich und mitunter müssen einige Systemparameter deshalb verändert werden. Es wird auch Plattenplatz für dazugehörende Dienstprogramme, Programmbibliotheken und Konfigurationsdateien benötigt, auch an eine eventuelle Vergrößerung von Page- und Swapbereichen des Betriebssystems ist zu denken, welche ja wiederum Plattenplatz benötigen.
- höhere Entwicklungskosten: Die Programmierung von X ist ziemlich komplex und aufwendig. Da das X-Protokoll eine allgemeine Client-Server-Schnittstelle darstellt, sind viele komplexe Strukturen notwendig, um diese Schnittstelle vollständig zu beschreiben und zu konfigurieren. Dabei ist vor allem zu beachten: Da die Applikation nicht an einen bestimmten Server gebunden ist, müssen dessen Eigenschaften vorerst abgefragt werden, sodann ist zu entscheiden, ob diese suffizient sind, um die Applikation sinnvoll zu betreiben. Wenn ja, sind entsprechende Strukturen aufzubauen und Konfigurationsparameter zu setzen, es muß sich die Applikation in gewissen Grenzen an die Fähigkeiten des Displayservers anpassen. Ferner müssen im laufenden Betrieb ständig "Eventqueues" abgefragt werden, auf Relevanz für das entsprechende Bildschirmfenster überprüft werden, und dementsprechend Aktionen gesetzt werden, oder irrelevante Events wieder zurückgestellt ("requeued") werden, sie könnten ja für ein anderes Fenster bzw. für eine andere Applikation von Bedeutung sein.(Schiebt man den Mauszeiger quer über den Bildschirm, so kann das schon eine beträchtliche Menge von Events auslösen.) Auch sind die Events wichtige Synchronisationsmöglichkeiten zwischen Client und Server - ansonsten völlig richtige X-Aufrufe können mitunter ihre Wirkung völlig verfehlen, wenn man diese Synchronisation außer acht läßt. Also, alles in allem viel Entwicklungsarbeit, viel Know-how und daher höhere Entwicklungskosten, mit einigen schon erwähnten Ausnahmen. Andererseits: je sorgfältiger man hier vorgeht umso größer sind wieder die Ersparnisse beim Portieren der entwickelten Software.

Schnittstellen:

Die unterste Zugangsebene zum X-Protokoll ist die sogenannte "X-lib". Die Verwendung dieser Schnittstelle ist sehr kompliziert und bedarf einiger Erfahrung und Know-how. Allerdings sind die X-lib-Aufrufe überall gleich und sie versprechen die leichteste Portierbarkeit. Mehr Möglichkeiten bietet das "Toolkit", welches bereits vorgefertigte graphische Objekte ("Widgets") anbietet. Damit geht's etwas leichter, die Portierbarkeit ist aber nur gegeben, wenn man sich auf die von MIT ausgelieferte Untermenge beschränkt, oder man den Quellcode für die Xtools-library der entsprechenden Firma besitzt (und mit portieren darf). Es gibt auch eine Reihe von Dienstprogrammen, welche zusammen mit selbstdefinierten Menüs (dazu muß man nur den Editor und die Syntax der Konfigurationsdatei des Windowmanagers beherrschen), an die verschiedene Programme und "shell-scripts" angekoppelt werden können, auch für den "Nicht-Guru" sehr brauchbare Möglichkeiten zur Arbeitserleichterung bieten.

Was ist beim Portieren von Software zu beachten?

Wie gesagt, ist man auf der X-lib - Ebene ja "völlig" kompatibel, weiters wird vorausgesetzt, daß der Quellcode aus standardmäßigem "C"-Code besteht und daß Unterschiede zwischen verschiedenen Betriebssystemen bekannt sind und berücksichtigt werden (z.B. "BSD" und "SysV" ...). Was leider gar nicht kompatibel ist, ist die meist verschiedene Hardware, und da kann man schon seine Überraschungen erleben. Man denke allein schon an das triviale Problem der "byte-order" d.h. die Anordnung und Adressierung der "Bytes" im Speicher. Darüber stolpert man sicher, wenn man z.B. Pixeldateien (wo Bilder in Form von Bildpunkten gespeichert sind) von DEC-Rechnern oder auch PC's auf Rechner mit 68xxx-Prozessoren überträgt und mit der dorthin portierten gleichen Software weiterverarbeiten will (da muß man nun doch eine kleine Softwareänderung vornehmen). Will man mit X-lib Pixelbilder darstellen, so gibt es dazu weniger Unterstützung als für Vektorgrafiken. Ist meine Software z.B. auf einer Workstation mit 8-Bitplanes (d.h. 256 mögliche Farben oder Grauwerte pro Bildpunkt) klaglos gelaufen, so heißt das nicht, daß dies auch auf einer 12-Bit Workstation der Fall ist, obwohl man es so annehmen möchte, da die 256 Möglichkeiten ja eine Untermenge darstellen. Soll dies funktionieren, muß man nach der Serverabfrage je nach der Anzahl der Bitplanes den Speicherbereich, von dem aus das Bild an den Server übertragen wird, dynamisch einrichten, sowohl was seine Größe als auch seine Adressierungseinheiten anbelangt. Damit hatte ich schon selbst Probleme, weil man ja dazu geneigt ist anzunehmen, eine so "allgemein gültige" X-lib wäre imstande, einem ein so triviales Problem abzunehmen. Aber wie gesagt, eine wirkliche Allgemeingültigkeit beschränkt sich ja nur auf das Protokoll. Abschließend kann zum Problem Portierung gesagt werden: Je sorgfältiger man bei der Softwareentwicklung vorgeht und speziell auf alle Eigenschaften aller potentiellen Server eingeht und diese im Programmcode berücksichtigt (was eine große Erfahrung mit dem derzeitigen, und auch zukünftigen technischen Stand aller möglichen Workstations erfordert), desto leichter wird die Portierung dieser Software vonstatten gehen.

Zusammenfassung:

Wenn wir nun alle hier erwähnten Vor- und Nachteile abwägen, können wir nun folgende Argumente für die Verwendung von "X" zusammenfassen:

- wir haben mehrere miteinander vernetzte Workstations oder
- wir wollen verschiedenste Applikationssoftware erwerben und wollen auch in näherer Zukunft aus einem großen Angebot auswählen können.
- wir haben vor, Workstations von verschiedenen Firmen zu kaufen, oder uns nicht auf Jahre auf eine Hardwarefirma festlegen zu wollen.
- wir wollen eigene Software entwickeln und einen möglichst breitgestreuten Markt damit erreichen.
- wir sind am Austausch von Software und Daten mit anderen wissenschaftlichen Institutionen interessiert, was sich vor allem durch die immer zahlreicher und effizienter werdenden Netzwerkdienste mehr und mehr anbietet.

Das alles gilt natürlich nur, wenn wir bereit sind, unsere Workstations auch entsprechend auszurüsten. Ein großzügig bemessener Arbeitsspeicher und genügend Plattenplatz lassen "X" erst wirklich gut funktionieren, ohne die allgemeine Rechenleistung einer Workstation stark zu beeinträchtigen. Stehen sehr wenig Mittel zur Verfügung und treffen die oben angeführten Punkte kaum zu, sollte man sich überlegen, ob man nicht auch mit einfacheren Mitteln zum Ziel gelangt (für eine einfache Menüsteuerung z.B. wäre "X" verfehlt, dafür tut's auch die "curses"-library). Meine Gedanken zum Thema "X" basieren auf meinen noch relativ bescheidenen Erfahrungen. Sollten sie auf Grund Ihrer Erfahrungen zu anderen Einsichten gekommen sein, würde mich (und sicherlich viele andere auch) das interessieren. Vielleicht sollte man einen Gedankenaustausch mit elektronischen Mitteln aufrechterhalten (z.B. "Notes-Konferenz"). Meine Mail-Adresse: EGH780::SELOS oder selos@tuvie.at

EUCLID-IS Version 2.1

Jadwiga Rogl

Anders als traditionelle Systeme, die nur Draht- oder Flächenmodelle unterstützen, bietet die Version 2.1 von EUCLID-IS durch die Integration von Volumen- und Flächenmodell und von parametrischen Techniken eine neue Generation fortschrittlicher Modellier-Methoden. Darüberhinaus liefert sie auch die notwendige Genauigkeit für physikalische Berechnung und die Fertigung. Bei gleicher Benutzeroberfläche erfolgt die Trennung zwischen 2D und 3D nur durch unterschiedlich ansprechbare Funktionen. Die zugrundeliegenden Datenstrukturen sind absolut gleich sodaß ein problemloser Übergang von 2D auf 3D und vice versa gewährleistet ist. Durch den integrierten 2D-Teil, entspricht EUCLID-IS den Erfordernissen der CAD-Entwurfspraxis, wo ein großer Bedarf an reiner 2D-Zeichnungserstellung besteht (ca. 80% aller Aktivitäten im Konstruktionsbereich). Leitgedanke bei der Auswahl der jeweiligen Entwurfsart ist: "Use 3D when useful and necessary - and 2D when sufficient."

2D/3D Übergang

Im "2D-Drafting-Modul" wird neben der traditionellen Arbeitsweise von 2D-Systemen (Definition von Views, Erzeugen geometrischer Elemente wie Punkt, Linie und Kurve sowie deren Weiterbearbeitung durch Trimmen, Split, Bemaßung etc.) auch die produktive Möglichkeit unterstützt, im "3D-Modeling-Mode" erzeugte 3D Modelle für die Zeichnungserstellung einzubringen. Dies kann auf zwei Arten erfolgen. Zum einen ist eine Referenzierung des 2D-Views auf das 3D-Modell möglich. Hierbei ist im 2D-View als Geometrie das 3D-Modell enthalten. Bei dieser Methode steht die Vervollständigung des Modells zu einer Zeichnung (Bemaßung, Schnitte, Zeichnungsrahmen) im Vordegrund. Eine geänderte Geometrie in der Zeichnung ergibt sich automatisch aus einer Änderung des 3D-Modells. Zum anderen kann das 3D-Modell in den 2D-View projeziert werden (wahlweise mit oder ohne Glätten von Kurven). In diesem Fall sind im 2D-View als Geometrie die 2D-Linien enthalten, die sich aus der Projektion des 3D-Modells ergeben. Vorteil dieser Methode ist, daß Linien einfach und schnell mit den typischen 2D-Funktionen wie Trimmen, Split, Limit etc. geändert und ergänzt werden können. Auch hierbei wird der Konstrukteur bei durch das System unterstützt, indem die Zeichnung nach Änderung des 3D-Modells in gewissen Grenzen automatisch aktualisiert werden kann.

Geometrische Modellbildung

EUCLID-IS kann jeden geometrischen Körper, ausgehend von einem Profil durch Rotation, Verschieben, Führen längs einer Leitlinie oder Umhüllen bzw. durch Addition, Differenz oder Durchschnitt (also Boole'sche Operationen) dieser 3D-Modell erzeugen. In EUCLID-IS werden Freiformflächen durch eine einzelne oder miteinander verbundene Bezier-Fläche dargestellt. Diese Flächen können leicht manipuliert werden (z.B. automatisches Trimmen mit logischem Limitieren) und zur Bildung komplexer Teile miteinander kombiniert werden. Die Handhabung von Freiformflächen wird durch Boole'sche und topologische Operationen erleichtert und verbessert, so daß wirklich jede denkbare Freiformfläche (einschließlich Löcher) darstellbar ist. Diese komplexen Freiformflächen können weiterhin mit allen in EUCLID-IS verfügbaren Volumenmodell-Funktionen bearbeiten werden. Somit ermöglicht EUCLID-IS Unterstüzung des gesamten Engineering-Prozesses mit einem einzigen einheitlichen Datenmodell, daß nachfolgende Anwendungen, wie FEM, die Berechnung physikalischer Eigenschaften und Programmierung von NC-Werkzeugwegen ebenfalls sicherstellt.

Freie Entwurfsprüfung, schnelle Netzgenerierung und Analyse

Aufgrund seiner Volumenmodell-basierten Teildefinition unterstützt EUCLID-IS auch entscheidende Analysefunktionen wie z.B. statische und dynamische Kollisionsuntersuchungen. So kann das Verhalten eines Teiles zuverlässig in einer komplexen, der "wirklichen Welt" entsprechenden Betriebsumgebung getestet werden. Mit dem Kinematik-Paket können Ingenieure komplexe, kinematische Strukturen interaktiv entwerfen. Nach der Festlegung von Position und Art starrer Teilkörper und derer Verbindung durch Gelenke, helfen leistungsstarke Simulations- und Editierwerkzeuge, die funktionale Architektur einer Struktur zu optimieren. Mit EUCLID-IS ist die Netzgenerierung ein unkomplizierter Vorgang. Der Ingenieur kann schnell und effizient direkt aus der Originalgeometrie des Entwurfsmodells ein Analysenetz erzeugen. Die Netzgeneratoren von EUCLID-IS ermöglichen die Definition von 1D- (linearen) 2D- (Oberflächen) und 3D-Elementen (Volumina). Eine Auswahl manueller und automatischer Konstruktionstechniken hilft, ein optimales Netz zu erzeugen, das den besonderen geometrischen und analytischen Anforderungen entspricht. Die Netzgenerierung kann bei EUCLID-IS völlig automatisiert werden. Durch die Verwendung von Dreieckselementen können automatisch Netze komplexer Oberflächen erzeugt werden. Ferner werden für jedes EUCLID-IS-Objekt automatisch Netze mit Volumenelementen generiert. Um die Netzgenauigkeit weiter zu erhöhen, erlauben leistungstarke Editierfunktionen intreraktiv zu arbeiten und so viele lokale Netzverfeinerungen wie nötig zu implementieren, bis die gewünschte Genauigkeit erreicht ist. Routinen wie Glätten, lokale Abstimmung der Netzdichte, Einfügen, Verschieben und Löschen von Knoten gestatten es ein geeignetes Netz zu erzeugen. EUCLID-IS simuliert interaktiv das Verhalten eines Teiles in einer realistischen, physikalischen Umge-

bung. Durch Eingabe ortabhängiger Analyseinformationen wie Lasten, Randbedingungen und Materialeigenschaften ist sichergestellt, daß Einzelteile und Zusammenbauten sich entsprechend der aufgestellten Entwurfskriterien verhalten. EUCLID-IS stellt hochwertige graphische Hilfen und mathematische Prüffunktionen zur Verfügung. Dies ist eine wichtige Hilfestellung um noch vor der Analyse die Genauigkeit und Vollständigkeit eines Netzes zu überprüfen. Funktionen wie die Kontrolle freier Kanten, Schrumpfen der Elemente und automatische Equivalenzbildung entlang Begrenzungen helfen Fehler rasch zu entdecken und Netze zu optimieren. Der Netzmodellierer ist kompatibel mit vielen Standard-Analysepaketen zur Festigkeits- und Wärmeanalyse. EUCLID-IS arbeitet mit marktüblichen FEM-Programmen wie MSC/Nastran, Ansys, Abaqus, Moldflow, Samcef, Supertab und Systus. Sobald die Analyse abgeschlossen ist, werden die Ergebnisse direkt am Entwurfsmodell dargestellt. Dies geschieht unter Verwendung einer Reihe interaktiver, grafischer Hilfen und 3D-Manipulationswerkzeugen. Funktionen wie Farbschattierung, Konturenbildung mit ISO-Werten, Gradienten-Tracing und die Darstellung deformierter Strukturen helfen, jeden Ergebnisparameter zu untersuchen. Die Ergebnisse könen zur Verdeutlichung dem Originalmodell überlagert werden.

Entwurf von Kunststoffteilen, zugehgörigen Formen, Spritzguß-Simulation und -Analyse

EUCLID-IS untersützt den Anwender in den einzelnen Phasen des Entwicklungszyklus eines Kunststoff-Produktes:

- Entwurf der Kunststoffteile und der Baugruppen
- Analyse der Form eines Teils
- Ableitung der Spritzgußform
- Simulation des Füllvorganges und Bestimmung der thermischen und mechanischen Bedingungen
- Entwurf der Spritzgußformen mit Normteilen
- Erzeugung der Werkzeugmaschinenbefehle für die NC-Fertigung der Form

Vom Design direkt zur Fertigung

Im Gegensatz zu vielen anderen CAD/CAM-Systemen beinhaltet EUCLID-IS nur eine kohärente Datenbank, in der vielfältige Informationen gespeichert und verwalten werden. Daten, die für den gesamten Fertigungsprozeß notwendig sind, können jederzeit abgefragt und geändert werden. Dies umfaßt Teildefinitionen, Maschinencharakteristika, verfügbare Werkzeuge sowie Technologieparameter, z.B. Eilgang, Arbeits- und Vorschubgeschwindigkeit, Toleranzen usw. Da die Bearbeitungsdaten direkt mit der Konstrunktionsgeometrie verknüpft sind, gestattet EUCLID-IS eine automatische und vollständige Anpassung der Werkzeugmaschinenwege, wann immer ein Teil modifiziert wird. Darüberhinaus wird ein großes Spektrum von NC-Bearbeitungsmöglichkeiten angeboten. Dazu zählen: Fräsen, Bohren, Drehen, Stanzen, Nibbeln, Drahterodieren und Laserschneiden. Alle von EUCLID-IS generierten Werkzeugwege können am Bildschirm dargestellt und sofort überprüft werden. Außerdem kann der Fertigunsingenieur interaktiv erkennen, wie sich Werkzeug und Werkzeugwege zu den Bearbeitungsteilen verhalten. Durch diese Simulation werden kostspielige Fräsversuche und Nachbearbeitungen vermieden.

Erweiterung von EUCLID-IS Version 2.1

Neben der vollständigen Integration eines 2D-Zeichnungs-Erstellungsteils weist 2.1-Version noch zahlreiche Verbesserungen und Erweiterungen auf:

- Reorganisation der Bildschirmaufteilung (dadurch bis zu 30% reduzierter Eingabeaufwand)
- Neue Identifizierungsfunktionen (z.B. Window)
- UNDO - Funktion
- Neu eingeführte LAYER-Technik (bis 1024)
- Macro-Command-Mode zum wiederholten Abarbeiten interaktiv erstellter Engabesequenzen mit geänderten Parametern.
- Optimierung der Visualisierung und ein bis zu 6 x schnelleres color shading
- Erhöhte Genauigkeit durch Codierung exakter Schnittpunkte und double precision-Verarbeitung bei bei Transformationen
- Erweiterte Datenstruktur
- Bessere Auswertung der Plattenkapazität durch dynamisch Anpassung der Datenbasengröße
- Einfachere Installation durch Benutzung von VMSINSTALL
- EUCLID-IS 2.1 läuft auf allen Rechnern ab VAXstation 2000.

Erstmalig ist an der Abt. Prozeßrechenanlage eine EUCLID Version, die mit allen Modulen ausgestattet ist, angeschafft worden. Nach Bewältigung organisatorischer Umstände (notwendiger Plattenplatz) und einer Testphase von ca. einem Monat wird dieses leistungsfähige Werkzeug den Benutzern zur Verfügung gestellt.

Chronologie des graphischen Entwurfssystems der Firma CALMA

Günter Houdek, Günther Wehrberger

1. Rückblick

Das graphische Entwurfssystem der Firma CALMA wurde bereits 1979 vom Institut für Allgemeine Elektrotechnik und Elektronik (Prof. Dr. Paschke) angeschafft, und die Abt. Prozeßrechenanlage übernahm die Betriebsführung dieses ersten CAD-Systems der **TU**-Wien. Die hohe Akzeptanz der Anlage in der Anfangszeit des computerunterstützten Konstruierens führte an der Abt. Prozeßrechenanlage schließlich durch das angesammelte Know-How zur schwerpunktmäßigen Betreuung der Institute im CAD-Bereich.

1.1 Einsatzschwerpunkte

1.1.1 Leiterplatten-Entwicklung

Das graphische Entwurfssystem der Firma CALMA war auf Jahre hinaus die einzige Möglichkeit an der **TU**-Wien, Leiterplatten für Lehre und Forschung (Diplom- und Seminararbeiten, Hardwareentwicklung für Projekte) zu entwerfen. Die ursprüngliche Konfiguration des Systems, die in Bild 1 dargestellt ist, bestand aus einem Doppelrechner-System mit Rechner-Hardware der Fa. Data-General, vier graphischen Arbeitsplätzen, und im wesentlichen aus zwei Softwarepaketen. Durch den Ankauf eines Photoplotters im Jahre 1982 wurde das CAD-System ideal ergänzt, da nun auch der fertige Film, die Vorlage für eine Leiterplattenproduktion, von der Abt. Prozeßrechenanlage zu erhalten war.

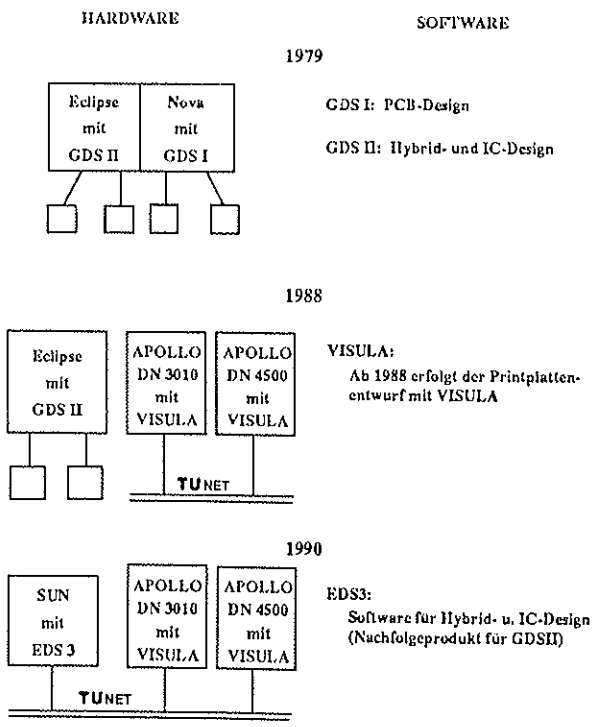


Bild 1: Chronologie des graphischen Entwurfssystems

Leiterplattenentwicklung (PCB-Design) wurde mit der Software GDS I auf der Data General NOVA betrieben. Dieses System war über Jahre hindurch überdurchschnittlich hoch ausgelastet (bis 18 Stunden pro Tag). Erst das Aufkommen leistungsfähiger PCs mit vergleichsweise billiger Software für PCB-Design führte zu einem Rückgang in der Auslastung. Trotzdem war das System für jene Institute, die aufgrund der finanziellen Situation, bzw. zu geringer Auslastung eines eigenen Systems, weiterhin die Anlage an der Abt. Prozeßrechenanlage benützten, von größter Wichtigkeit.

1.1.2 Entwurf integrierter Schaltkreise

Der Entwurf integrierter Schaltkreise (IC-Design) oder die Entwicklung von Hybridschaltungen erfolgte über mehr als zehn Jahre mit dem Software-Paket GDS II und der Data General ECLIPSE S230. GDS II war als ausgereifte und nach wie vor leistungsfähige Software für High-Tech-Anwendungen prädestiniert, und bot den Fachinstituten Eigenschaften, die im PC-Bereich nicht angeboten werden.

1.2 Ersatz des CALMA-Systems

Die Mitteilung der Firma CALMA, daß der Hardware-Wartungsvertrag für das besonders veraltete NOVA-System nicht mehr aufrecht gehalten werden kann, war der Anlaß, daß 1988, wieder gemeinsam mit dem Institut für Allgemeine Elektrotechnik und Elektronik (Prof. Dr. Zach), ein dem Stand der Technik entsprechendes CAD-System für die Leiterplattenentwicklung angeschafft wurde. Das System, bestehend aus der VISULA-Software von Racal-Redac, und einer Apollo DN3010-Workstation, war bald so stark ausgelastet, daß im Sommer 1989 ein zweiter VISULA-Arbeitsplatz mit einer Apollo DN4500 angekauft wurde. Damit stehen am EDV-Zentrum wieder zwei Arbeitsplätze mit moderner Hard- und Software für die Leiterplattenentwicklung zur Verfügung, die zwar in erster Priorität von der Abt. Leistungselektronik genützt werden, die freien Kapazitäten sind aber groß genug, um auch den Bedarf der anderen Benutzer abzudecken.

Der Entwurf integrierter Schaltkreise wäre nach wie vor mit der Software GDSII möglich, ist aber aufgrund der hohen Wartungskosten für die mehr als zehn Jahre alte Data General ECLIPSE S230 wirtschaftlich nicht mehr vertretbar gewesen. Aus diesem Grund wurde im vergangenen Jahr auch ein Ersatz für den noch verbliebenen Rest des ersten CAD-Systems der **TU**-Wien gesucht. Da die Benutzer eine funktional gleich gute Software wie GDS II forderten, die auch das Weiterverarbeiten von GDSII-Datenbeständen ermöglichen sollte, kam nur EDS 3, das

Nachfolgeprodukt von GDS II in Frage. Die IC-Softwareentwicklung wurde in der Zwischenzeit von der Firma Calma an die Firma VALID abgegeben, welche in einem neuen Produkt namens CONSTRUCT, EDS 3 und den eigenen Layout-Editor gemeinsam anbietet. Da diese Software derzeit nur auf Hardware von SUN lauffähig ist, wurde zu Beginn dieses Jahres eine SUN-Workstation der oberen Leistungsklasse mit einer Testversion der EDS 3-Software an der Abt. Prozeßrechenanlage in Betrieb genommen.

2. IC Layout Editor Construct auf SUN/Sparc 330:

Ab Mai 1990 wird das neue Produkt Construct der Fa. VALID am EDV-Zentrum angeboten.

2.1 Produkt-Beschreibung:

Construct ist ein hierarchischer Integrated Circuit (IC) Layout Editor, speziell zur Erstellung von strukturierten, kundenspezifischen Designs. Dieses Paket enthält eine Fülle von erweiterten Editierungsmöglichkeiten für IC-Design-Prozesse und -Technologien. Der Befehlssatz ist für einfache Handhabung ausgelegt, und beinhaltet auch wahlweise das in der Industrie als Standard geltende GDS II-Konzept. Dadurch wird das Training auf diesem neuen Produkt auf ein Minimum gehalten.

2.2 Eigenschaften von Construct:

- erweiterte Editier- und Layoutfähigkeiten
- unterstützt alle gängigen IC-Entwurfsmethoden und Technologien
- GDS II-ähnliche Command-Syntax und Konzept
- 32 bit Datenbasis (all-angle)
- Multiaktive Viewports (siehe Bild 2)
- flexible Benutzerschnittstelle
- On-line boolsche Operationen
- leistungsfähige Programmiersprache EPL mit GPL- (GDS II kompatible Programme) Translator
- integrierte Entwurfsumgebung mit 'shared database'

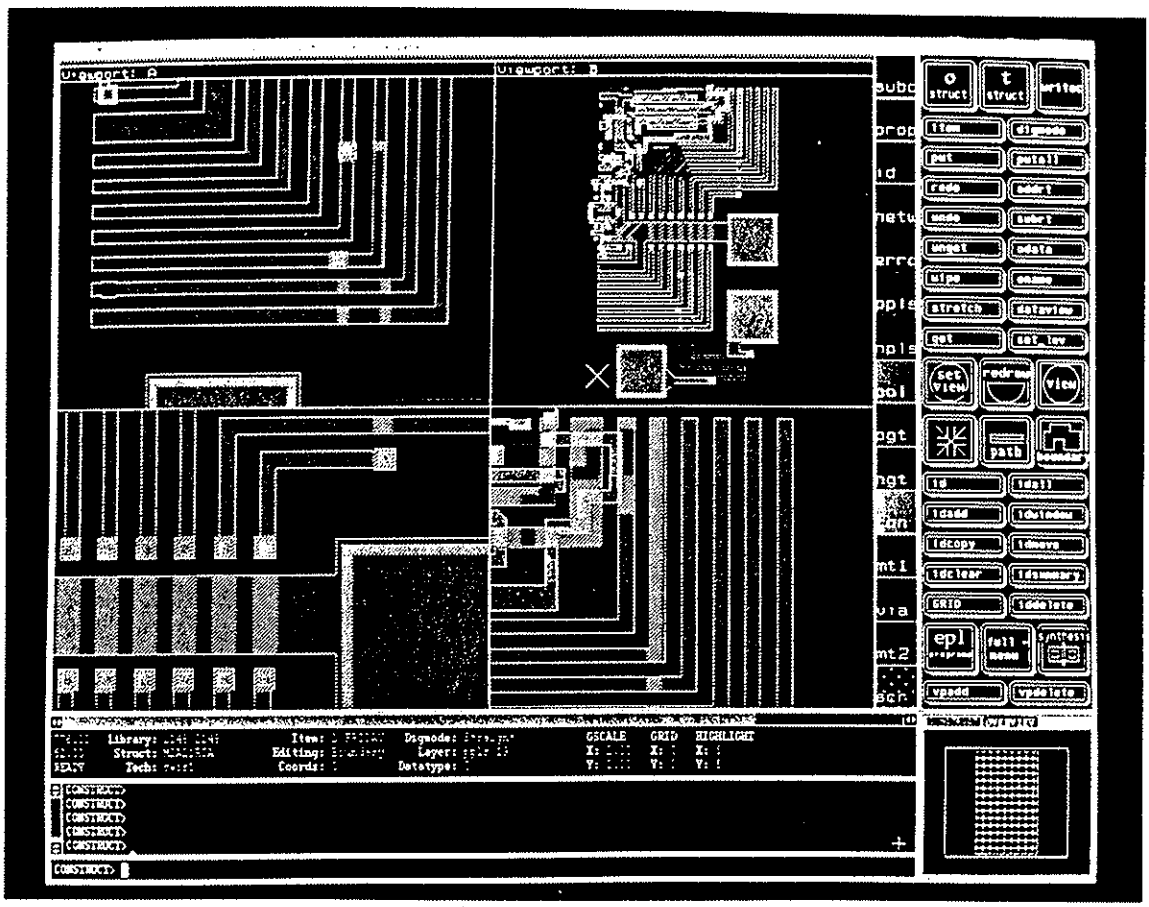


Bild 2
Multiaktive Viewports

Es können bis zu 26 Viewports gleichzeitig dargestellt und editiert werden. Geometrien können von einem Fenster in das andere kopiert, sowie Bahnen die in einem Fenster begonnen wurden in einem anderen fortgesetzt werden. Die Befehle werden zwischen den einzelnen Fenstern (Viewports) übertragen und automatisch erkannt, wodurch der Designaufwand wesentlich reduziert wird.

Flexible Benutzeroberfläche:

Wie schon vom GDSII her bekannt, können die Menüs individuell umkonfiguriert werden. Weiters stehen Macros, Pop-up Menüs, Icon Button Menüs und Funktionstasten zur Verfügung, oder die Befehle werden per Tastatur eingegeben. Dabei können die Kommandos abgekürzt verwendet werden (siehe Bild 3).

On-line logische Operationen:

Construct erlaubt das Kreieren von Geometrien mit beliebigen Winkeln (32-bit Database) für jeden Technologietyp wie z.B. Bipolar oder GaAs. Komplexe Geometrien können einfach durch boolsche Operationen (AND, OR, XOR, ANDNOT) sowie 'oversizing' und 'undersizing' erzeugt werden.

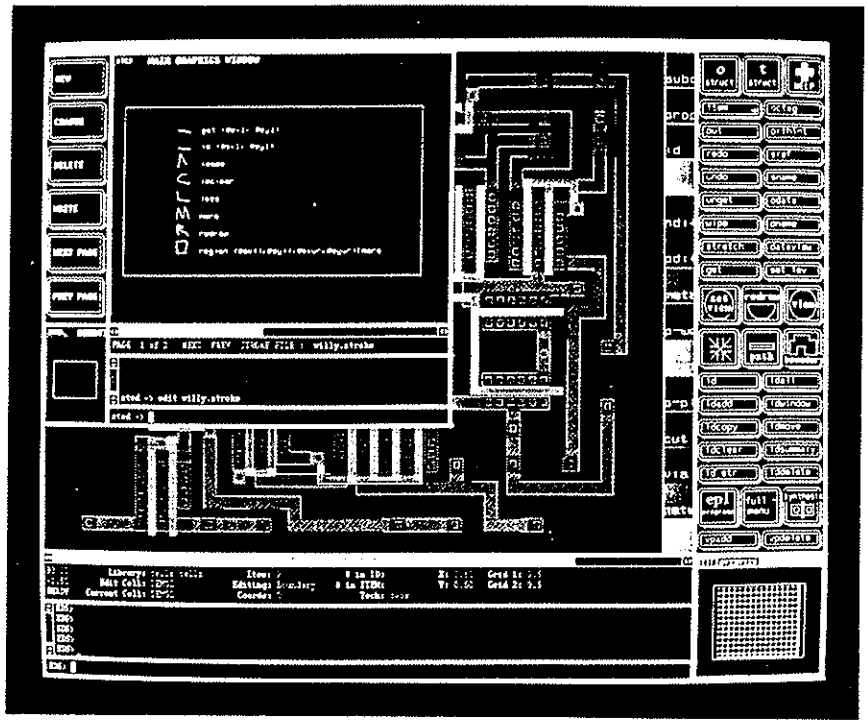


Bild 3
Definieren von neuen Kommandos und zeichnen von neuen Menüs, während die Struktur angezeigt wird.

Programmierbare Strukturen:

Programmierbare Strukturen sind Benutzer-definierte Programme, die automatisch Strukturen wie Transistoren, Widerstände, Gatter usw. produzieren, welche dann manipuliert und gedehnt werden können, um die vorgegeben Design-rules einzuhalten. Ein spezielles Interface ermöglicht Nicht-Programmierern solche PCELLS interaktiv am Schirm zu entwickeln. Von dieser erstellten Graphik wird automatisch das Programm erstellt. Wird das Programm modifiziert, so werden durch Aufruf dieses Programms die gewünschten Strukturen 'upgedated'.

EPL Programmier-Sprache

EPL ermöglicht Entwicklern ein Automatisieren von Designabläufen, sowie kreieren von eigenen Kommandos. Innerhalb EPL können alle Construct-Befehle verwendet werden, sowie eingebaute EPL-Funktionen, mathematische Funktionen, und all jenen Eigenschaften, die man von den üblichen höheren Programmiersprachen kennt. EPL's 'State of the Art' strukturierte Programmiertechniken ermöglichen eine schnelle Funktionserweiterung von Construct.

GDS II-Kompatibilität:

Diese Kompatibilität ist sowohl auf Database- als auch auf Kommandoebene gegeben. Weiters können bestehende GPL-Programme auf die neue EPL-Sprache konvertiert werden.

Offene Struktur:

Auf Grund der offenen Strukturen ist eine einfache Portierbarkeit zu anderen bestehenden Layouteditor-Programmen gegeben. Es werden folgende Industriestandard Formate unterstützt:

- GDS II Stream Format
- CIF (Cal Tech Intermediate Format)
- Patterngenerators (e-beam & optical)
- Pen- & Photoplotters

Für weitere Auskünfte steht D.I. Houdek gerne zur Verfügung.

Erstellung von Software als "mehrdimensionales" Problem, eine alternative Betrachtungsweise

Antonin Sprinzl

"Gott: Ich will eine Welt machen.
Wissenschaftler: Welche Welt ?
G: Die Welt.
W: Die Welt gibt es nicht. Kant und die Wissenschaftstheorie werden es beweisen. Es gibt nur Deutungen der Welt.
G: Ich kenne alle Deutungen.
W: Warum sich dann nicht mit den Deutungen zufriedengeben ? Die Welt wäre entbehrlich.
G: Ich will eine Leonardo-Welt, keine Leibniz-Welt.
W: Mach' sie. Sie wird dieselbe sein."
(aus J. Mittelstraß, "Der Flug der Eule", von der Vernunft der Wissenschaft und der Aufgabe der Philosophie, suhrkamp 1989)

Zusammenfassung

'Software-Erstellung', 'Software Engineering' sind im Rahmen der informationstechnologischen Entwicklung zu Reizbegriffen geworden, sogar von Krisen ist die Rede. Woran mag das wohl liegen ?

Der Bedarf nach neuartigen, immer komplexeren IT-Systemen ist im Steigen begriffen. Die Erstellung zeitgemäßer Software ist zu einer nicht trivialen Angelegenheit geworden. Die beträchtliche Auslegungsbreite einschlägiger Begriffe und Sprachkonzepte scheinen zur Verschärfung der Problematik ihren Beitrag zu leisten. Die Überbetonung des "Instrumentellen" sowie die Verharmlosung des "Faktors Mensch" im Vorhaben der Software-Erstellung sind mit vielen Fragezeichen versehen.

"Software Engineering" als eine eigenständige Ingenieurdisziplin blickt auf eine kurze "Tradition" zurück.

Einleitung

Der stürmische technologische Fortschritt auf dem Gebiet der Informationstechnologie scheint langsam, zumindest was im konkreten die Hardware- und Software-Entwicklung anlangt, gewisse Konturen anzunehmen.

Die Entwicklung auf dem Gebiet der Hardware vermittelt den Eindruck, einen gezielten, systematischen Weg zu beschreiten, kontinuierlich begleitet von sichtbaren Erfolgen. Die Leistungssteigerung liegt, trotz allen pessimistischen, eine baldige Sättigung prophezeihenden Prognosen, weiterhin in der Größenordnung einer Zehnerpotenz alle 3 bis 5 Jahre. Das Geschehen auf dem Gebiet der Entwicklung der Software erweckt hingegen bei weitem nicht den Eindruck eines ähnlich erfolgreichen, durch "klare Zielvorgaben" geprägten Weges. Die "Leistungssteigerung" liegt in einem kaum erwähnenswerten Bereich.

Deshalb werden im Laufe der letzten Jahre vermehrt Anstrengungen unternommen, "Software Engineering", diese amorph erscheinende Festung, von unterschiedlichen Seiten her "einzunehmen". Oft ist zu hören, das Problem sei "komplexerer" Natur. In Anlehnung an diese Metapher möge auch der folgende Beitrag verstanden werden. Es wird hier der Versuch unternommen, durch eine "Attacke" in Form einiger Gedankenskizzen die Problematik der Erstellung von Software, von einem alternativen Standpunkt aus, in Angriff zu nehmen. Der Themenkreis der "besten" Methode, des "besten" Tool usw. für die Erstellung von Software, der oft genug in den einschlägigen Publikationen behandelt wird (und nicht selten dem "Religiösen" zuzurechnenden ist), soll hier nicht näher diskutiert werden (Literaturhinweise vorhanden).

In einschlägigen Kreisen besteht ein allgemeiner Konsens darüber, welche Bedeutung der Problematik der Erstellung von Software heute zukommt bzw. in welcher Situation sich dieses Gebiet gegenwärtig befindet. Zum einen findet die im Steigen begriffene Komplexität von IT-Systemen zunehmend ihren wesentlichen Niederschlag in der immer umfangreicher ausfallenden System-Teilkomponente Software, die bestimmten qualitativen Forderungen genügend, in einem ökonomischen Rahmen erstellt werden soll, so weit zum Anliegen und seiner Relevanz. Zum anderen mangelt es weiterhin, abgesehen von einigen partikulären Ansätzen, an ausreichend allgemeinen und "leicht" erlernbaren, anwendbaren, prüfbaren Konzepten und durchschaubaren Verfahren, die darüber hinaus durch geeignete Tools unterstützt werden, so weit zum Problem.

Manchmal kann der geäußerte Wunsch vernommen werden, Software ähnlich wie andere Produkte, factory-, industriemäßig (am besten gleich einem "Schema" folgend) erzeugen zu wollen, was natürlich die Realisierung einer ausreichend klar formulierten Absicht impliziert. Kann das Paradigma der industriemäßigen Erstellung von Produkten für das Unterfangen "Software-Erstellung" mit ihren meistens verschwommenen Vorstellungen zielführend herangezogen werden ?

Im folgenden wird versucht, aus heutiger Sicht, von einer alternativen Warte aus betrachtet, einige wenige potentielle Problembereiche der Software-Erstellung in "Skizzen" näher zu umreißen. Überlegungen berühren primär softwaremäßige Komponenten eines IT-Systems *(Hardware- und organisationsspezifische Aspekte stehen hier nicht im Mittelpunk)*.

Die Auswahl beschränkt sich auf folgende Bereiche:

- Realität, Abstraktion
- Darstellung
- Mensch, Automatismen
- Instrumentarium
- Ausdrucksmittel
- Weltentrennung
- Transformationsprozeß
- Software als finales vs. evolutionäres Produkt

Erstellung von Software, Zielsetzung, "didaktische Arbeitsskizze"

Den weiteren Überlegungen möge in einer Übersichtsskizze das Wesen der Software-Erstellung vorangestellt werden.

Bei diesem Vorhaben geht es im wesentlichen um die (ökonomische, ingenieurmäßige) Schaffung eines geeigneten technischen Artefaktes *(primär Kollektion ausführbarer Programme sowie begleitende Doku)* einer avisierten Problemstellung, einer Idee, eines Realitätsausschnittes, dessen "Einbettung" in eine reale Umgebung uns die Illusion vermitteln soll, sich durch die gleichen uns relevant erscheinenden qualitativen, quantitativen Eigenschaften auszuzeichnen wie die Problemstellung, die Idee, der Realitätsausschnitt selbst, von der (dem) ausgegangen wurde.

Besonders hervorgehoben sei die Tatsache, daß wir im Rahmen der Erstellung von Software mit zwei globalen Problemkategorien zu tun haben,

- mit einer **produkt**-orientierten Kategorie von **was**-Problemen ("was soll erstellt werden ?") und
- einer **prozeß**-orientierten Kategorie von **wie**-Problemen, die die unterschiedlichen Aspekte der **Erstellungsprozesse** (Planung, Management, Kontrolle) enthält.

Diese zwei Kategorien sind auf vielfältige Weise miteinander verwoben.

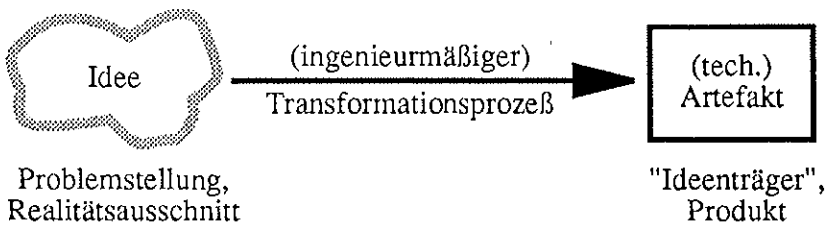


Abb. 1 Software-Erstellung, Arbeitsskizze

Für einfache, ausbildungsorientierte Problemstellungen mag dieses Grundkonzept der Software-Erstellung durchaus genügen. Für die reale Praxis ist dieses Konzept jedoch ungenügend.

Bei näherer Betrachtung der Übersichtsskizze können zumindest folgende globalen Problemkreise herausgeschält werden:

a) "Erkennen", "Eingrenzen" des Realitätsausschnittes,

b) "Verkörpern" des Realitätsausschnittes,

c) "Korrektur" des Realitätsausschnittes (erkenntnisbedingt).

Ein kurzer Rückblick auf die bisherigen Praktiken der Erstellung von Software läßt leicht die Feststellung zu, daß die Bedeutung des Problemkreises ...

- des Verkörperns überbewertet,
- des (systematischen) Erkennens weitestgehend unterbewertet,
- der erkenntnisbedingten Korrektur in seiner vollen Tragweite nicht erkannt

wurde. Insbesondere im Lichte der Tatsache, daß

- komplexe Problemstellungen eine Umsetzung (Erstellen, Betreiben) in pluraler Umgebung inkludieren,
- der Mensch letztendlich ein erkenntnisorientiert veranlagtes, einer Evolution unterworfenes Wesen ist,

gewinnt der Problemkreis des **Erkennens** und der wiederholenden **Korrektur** des Realitätsausschnittes zunehmend an Bedeutung.

Die wesentliche Aufgabe des **Software Engineering** (SE) im engeren Sinn besteht vermutlich darin, den Transformationsprozeß unter Einsatz eines bewährten, theoretisch fundierten Instrumentariums (Techniken, Methoden, Verfahren), praktisch und ökonomisch durchzuführen.

Läßt sich aber diese Tätigkeit aus dem Gesamtkomplex der Software-Erstellung leicht herauspräparieren ? Kann es klare Trennlinien geben, wo beginnt und endet das Erkennen, das Verkörpern, die Korrektur ?

Problembereich "Realität, Abstraktion"

Es ist stets wichtig, Realität und (realitätsabbildende) Abstraktion als zwei unterschiedliche Metakonzepte, auseinanderzuhalten. (Der Begriff "Realität" soll hier auch Begriffe wie Realitätsausschnitt, Fachdomäne, Domänenausschnitt, Problemstellung vertreten.)

Eine Realität stellt für die menschliche Interpretierbarkeit (und Empfindungen) eine stets potentiell unendliche Quelle dar. Infolgedessen wird eine gegebene Realität aufgrund des unterschiedlich ausgeprägten Wahrnehmungsvermögens der Menschen "intern" in unterschiedlichen abstrakten Bildern (Abstraktionen) festgehalten. Jeder Mensch "sieht" eine gegebene Realität mit unterschiedlichen Präferenzen, höchst individuell. Zur individuellen Ausprägung dieser Sichtweise haben, wie allgemein bekannt, viele Lebensfaktoren entscheidend beigetragen, wie z.B. kultureller, sozialer, politischer, wirtschaftlicher Background, Bildung, einschlägige Ausbildung, erworbene Erfahrungen, u.v.a.m.

Menschen kommunizieren ihre "Realitätsempfindungen" untereinander zum erheblichen Teil mit Hilfe von geeigneten Abstraktionen. (Auf sonstige Kommunikationsformen wird hier nicht näher eingegangen). Abstraktionen als Ideenmanifestanten ermöglichen einen qualifizierbaren Diskurs, einen Meinungsvergleich.

Eine Abstraktion kann als ein auf eine endliche Dimension skalierter, relevante Züge manifestierender "Abzug" eine gegebene Realität nie vollständig erfassen. Deshalb kann sie bestenfalls als ein reduziertes (verzerrtes, sehr oft "verunstaltetes") Abbild einer viel "reicheren" Realität angesehen werden. Wichtige Merkmale, die eine Abstraktion kennzeichnen, sind ihre Abbildungstreue und Schärfe. Sowohl die Wahl der konstituierenden Wesenszüge als auch die Ausprägung der Abbildungsschärfe gehören in den Bereich einer höchst individuellen Festlegung.

Je nach Komplexität der Sachverhalte und nicht zuletzt durch geschichtliche Entwicklung bedingt, erfahren manche Abstraktionen bzgl. der Übereinkunft ihrer Realitätsentsprechung eine gewisse "Stabilität". Andere Abstraktionen hingegen können sich im Zustand einer kontinuierlichen Ausgrenzung und Präzisierung befinden. Eine Problemstellungs-Novität etwa wird infolge der Interessensvielfalt aller Betroffenen einen Anlaß zu mehreren Realitätsauslegungen bieten. In einer solchen Situation ist die "Erarbeitung" einer adäquaten, allgemein anerkannten Abstraktion eines der zentralen Probleme (man denke an eine Software-Erstellung *(etwa die Konstruktion eines optimalen(?) Integrated Project Support Environment als Software Engineering Tool)* mit vielen Mitwirkenden *(Auftraggeber, Geldgeber, System-Benutzer, Domänenfachleute, Theoretiker, Analytiker, Programmierer, Verwalter, Organisatoren, Qalitätsprüfer, Wartungsfachleute, u.a.m.)*).

Das Vorhandensein geeigneter Abstraktionen bzgl. was und wie ist für eine zielführende Software-Erstellung jedoch unerläßlich.

Problembereich "Ausdrucksmittel"

Um Abstraktionen "sichtbar" zu machen sowie diese einer weiteren Behandlung zu unterziehen, sie **operationalisieren** zu können, bedarf es geeigneter Ausdrucksmittel (Bild, Sprache, ...). Eine ökonomische Darstellung von Abstraktionen erfolgt derzeit überwiegend durch eine symbolische Sprache, die eine Automatismen-Manipulierbarkeit gestattet. Die Sprache stellt den "Träger" von Abstraktionen dar. Sie sollte sich durch eine Fülle qualitativer Eigenschaften auszeichnen, die zum Teil einander widersprechender Natur sind. Zu diesen gehören Mächtigkeit, Ausdruckskürze und Prägnanz, leichte Erlernbarkeit, Verständlichkeit (wünschenswerte Modelleigenschaften), Vollständigkeit, Konsistenz, algorithmische Manipulierbarkeit, Prüfbarkeit (wünschenswerte Formalismeneigenschaften), u.a.m.

Der Operationalisierungsaspekt der Ausdrucksmittel ist hier von zentraler Bedeutung. Dies betrifft natürlich beide Teilbereiche der Software-Erstellung, sowohl den **produkt-** als auch den **prozeß-**orientierten Bereich. Zur Operationalisierung von Abstraktionen (Wahl geeigneter Ausdrucksmittel) folgen nun zwei Bemerkungen.

Bemerkung 1

Bei der Operationalisierung von Abstraktionen im produktorientierten Bereich z.B. (die Überlegungen beziehen sich natürlich auch auf den prozeßorientierten Bereich) besteht die Problematik zuerst darin, einen Mittelweg zwischen folgenden Extrema zu finden, bei denen ...

a) ("menschenfreundlicher Extremfall") ein hohes Abstraktions-Niveau der Ausdrucksmittel (high description language) zwar wünschenswert (Verständlichkeit) ist, der "Preis" dafür aber in der erheblichen Schwierigkeit der Festlegung der Semantik, Prüfbarkeit, usw. liegt,

b) ("maschinenfreundlicher Extremfall") niedriges Abstraktions-Niveau der Ausdrucksmittel (machine language) für Konzeptualisierung zwar ungeeignet ist, dafür aber problemlose semantische Festlegung, Automatisierbarkeit, usw. gestattet.

Bemerkung 2

Da wir nun unsere Abstraktionen mit Hilfe von Maschinen zu operationalisieren beabsichtigen, wäre die Wahl des Extrems b) günstig. Das größte Hindernis der beschränkten Anwendbarkeit der Variante b) liegt in der

menschlichen Unfähigkeit, **komplexe** Sachverhalte auf einem derartig **niedrigen** Abstraktionsniveau zu erfassen und für die Maschine aufzubereiten.

Dies führt uns zum weiteren Problem, dem der Abstraktions-Transformation *(Im weiteren eine Einschränkung auf Probleme, die nur mit einer "top-down"-Transformation, d.h. einer Transformation von einem hohen auf ein niedriges Abstraktionsniveau zusammenhängen.)*. Es handelt sich in der Regel um einen mehrstufigen Abstraktions-Verfeinerungsprozeß *(Es ist dabei natürlich klar, daß eine Abstraktion im Zuge des Verfeinerungsprozesses, um weitere, der Operationalisierung letztendlich "zugute" kommende Details "angereichert" wird, deren Umfang von der jeweiligen Abstraktionsstufe abhängen.)* (der auftretenden Abstraktionskomplexität wegen), der durch den Einsatz weiterer, von der jeweiligen Abstraktionsstufe abhängiger Ausdrucksmittel begleitet ist. Wir haben es hier also u.U. mit einer Fülle von Ausdrucksmitteln zu tun, die unter Berücksichtigung unterschiedlicher Standpunkte und Bedürfnisse, stets den gleich bleibenden Sachverhalt (Abstraktion) zu präzisieren trachten.

Das Problem liegt nun nicht nur in der Wahl eines Ausdrucksmittels, das sich für die Behandlung spezifischer Zusammenhänge jeweiliger Abstraktionsstufe gut eignet. Es muß auch auf eine gewisse "Kongruenz" der Ausdrucksmittel untereinander geachtet werden, damit der Abstraktionsverfeinerungsprozeß womöglich inhaltsgetreu, "sprunglos" erfolgen kann.

Die Umgangssprache bietet einen weiteren Angriffspunkt. Eines der größten Probleme der gegenwärtigen Praxis scheint in der gewaltigen Fülle und Auslegungsbreite einschlägiger zum Teil neu geprägter Konzepte und Begriffe zu liegen, die im Zusammenhang mit Software-Erstellung auftreten. Es überrascht dabei natürlich nicht, daß die angebotene Auslegungsvielfalt sehr stark von der individuellen praktischen Erfahrung abhängt.

Das Nichtvorhandensein eines reichhaltigen, stabilen, semiotischen "Werkzeugkastens" wird jedoch ein weiteres Vorankommen in der Erschließung bedarfsorientierter Erstellungskonzepte weiterhin erheblich hemmen.

Problembereich "Darstellung"

Abstraktionen werden mit geeigneten Ausdrucksmitteln mitteilbar, "(be)greifbar", operationalisierbar gemacht. In diesem Zusammenhang erhebt sich nun die wichtige Frage nach einer geeigneten Darstellbarkeit und Visualisierbarkeit. Einer Darstellbarkeit, mit der sich vor allem der Mensch, der "Schöpfer" der Abstraktionen, anfreunden kann.

Wie allgemein bekannt, besitzt der Mensch die erstaunliche Fähigkeit, visuell aufgenommene Informationsinhalte, die in der Regel komplexer Art und Struktur sind, mit einer kaum zu überbietenden Effizienz mit Bedeutungen zu versehen, zu manipulieren und zu verarbeiten. Eine geeignete, bildhafte Darstellungsform der hier in Frage kommenden Abstraktionen wäre natürlich äußerst wünschenswert, die insbesondere ...

- die menschlichen, kognitiven Fähigkeiten in hohem Maß berücksichtigt und ihm eine natürliche Manipulierbarkeit $Dazu gehören Operationen wie das Einprägen, Merken und Abrufen, Zerlegen, Zusammenfügen, Erkennen von Mustern, Zusammenhängen, Inkonsistenz, Unvollständigkeit, u.a.m.$ abgebildeter Abstraktionen gestattet,
- oben angeführte, relevante Merkmale einer Sprache trägt.

Eine derartige Darstellungsform ist aber (zumindest bis auf weiteres) nicht bekannt. Daher werden wir auch weiterhin mit dem Problem der "Unsichtbarkeit" zu tun haben. Es ist (derzeit) unmöglich, eine Abstraktion komplexen Sachverhaltes so darzustellen, daß man ihre "Angemessenheit", "Fehlerfreiheit", oder aber "Unzukömmlichkeit", usw. sozusagen unmittelbar aus ihrer Darstellungsform, (qualitativ) beurteilen kann. (Würde man z.B. bei der Konstruktion des Modells einer Brücke das linke Geländer "vergessen", so wäre eine solcher "Mangel" sofort sichtbar. Im Falle eines technischen Artefaktes hoher Komplexität von der Art "Software" ist das Fehlen einer vergleichbar wichtigen Software-Komponente nicht unmittelbar zu ersehen). Dies betrifft natürlich auch die Unmöglichkeit der Visualisierbarkeit möglicher Operationsräume, in denen sich Abstraktionen "bewegen" können.

Historisch entwicklungsbedingt überwiegt derzeit eine nichtbildhafte Darstellung von abstrakten Sachverhalten, mit der sich "vielleicht" die Automatismen, nicht so rasch aber, wie die Erfahrung zeigt, der Mensch anfreunden kann.

Problembereich "Weltentrennung"

Es sei hier im besonderen auf die Wichtigkeit der klaren konzeptuellen Trennung der folgenden "Spezifikationswelten" hingewiesen:

- Spezifikation des Domänenproblems (Welt1) und
- Spezifikation des dazugehörigen Artefaktes (Welt2).

(Die von Popper übernommenen Begriffe Welt1 und Welt2 sind hier nicht im Popperschen Sinn in Verwendung.)

Als Beispiel für Welt1 können wir etwa die Fachdomänen Verkehrswesen, Software-Erstellung (hier als Domäne), Medizin, usw. betrachten. Als Beispiel für Welt2, die Realisierungsdomäne, nehmen wir das für uns hier

relevante Software Engineering in derzeit geläufiger, weitgefaßter Bedeutung (andere Ingenieurdisziplinen wären hier aber ebenso denkbar).

Vom Standpunkt einer sinnvollen Bewältigung des SW-Erstellungsvorhabens wäre naheliegenderweise folgende Vorgangsweise optimal: Welt1 liefert die Definition der Problemstellung (als W1-Modell, Domänen-Modell), Welt2 legt ihre praktische Realisierung eindeutig fest (als W2-Modell, SW-System-Modell, 'blueprint', eindeutige, erschöpfende Konstruktionsanweisung für deren Erstellung). Fertig. Praktisch ist leider dieses Konzept nicht allgemein anwendbar und daher von beschränkter Brauchbarkeit.

In realen Situationen, wie sich stets zeigt, liefert erst das Artefakt der Welt2 eine graduelle (erhebliche) Bestätigung (Widerlegung) der Brauchbarkeit der von der Welt1 definierten Vorgaben, was normalerweise eine Korrektur (Redefinition) des W1-Modells der Welt1 nach sich zieht (ziehen sollte). Dies kann, in Abhängigkeit von der Problemstellungs-Komplexität mehrere Iterationen zwischen den zwei Welten erfordern. Die zentrale Frage, die sich dabei ergibt, betrifft dann eine praktische, "ökonomische" Durchführung dieser Iterationen.

Bezüglich der gegenwärtigen Praxis hat man häufig mit einer Situation zu tun, in der ein in der Regel in der Welt2 "beheimatetes" Software-Team die relevanten Anliegen und Zusammenhänge der an sich "fremden" Welt1 zu erfassen hat. Dabei tritt das häufig anzutreffende Problem der klaren konzeptuellen Trennung der zwei Welten auf. Anliegen der Welt1 werden aus der Welt2-Perspektive, sozusagen durch eine "Welt2-Brille" der sich in dieser Welt "**offensichtlich**, unmittelbar" anbietenden "eleganten" Lösungen betrachtet. Es überrascht wenig, daß in solchen Situationen daraus unbefriedigende Beurteilungen oder sogar falsche Einschätzungen der in der Domänenwelt (Welt1) liegenden Sachverhalte resultieren. Dies kann auf mannigfache Gründe zurückgeführt werden (generell mangelnde Kenntnis der "fremden" Welt, "ausgeprägte", persönliche Blick-Präferenzen, usf.).

Möglichst genaue, erschöpfende Erfassung der Problemstellung samt einer eventuellen Estimation künftiger "Entfaltungstendenzen" ist für den Gesamterfolg der Software-Erstellung von eminenter Bedeutung. Es ist einsichtig, daß eine u.U. weniger effiziente Realisierung einer befriedigend eingegrenzten Problemstellung von größerer Brauchbarkeit ist als eine elegante, sehr effiziente Realisierung einer unbefriedigend erfaßten Problemstellung.

Problembereich "Mensch, Automatismen"

Unser Zeitalter trägt die Prägung einer Automatismeneuphorie. Diese Tendenz macht sich auch auf dem Gebiet der Software-Erstellung bemerkbar. Manchmal könnte man den Eindruck gewinnen, auch hier Menschen in nicht allzu ferner Zukunft durch geeignete Automatismen "beinahe" ersetzten zu können. Mit Hilfe von Automatismen lassen sich natürlich nur Aufgaben bewältigen, die letztendlich spezifizierbar sind.

Abgesehen von Trivialfällen, ist das Vorhaben der Software-Erstellung im allgemeinen als eine multi-agent multi-activity Angelegenheit zu werten. Alle Anzeichen sprechen dafür, daß eine nicht triviale Software-Erstellung infolge auftretender und nach oben zumindest potentiell nicht beschränkter Komplexitätsvielfalt sowohl auf der Produkt- als auch auf der Prozeßebene, nur in Sonderfällen, mit einem vernünftigen Aufwand erschöpfend spezifiziert werden kann. Es bestehen berechtigte Gründe zu der Annahme, daß der Mensch mit seiner Phantasie und seinen spezifischen Qualitäten und einmaligen Eigenschaften noch stärker als bisher eine zentrale Rolle im künftigen Erstellungsprozeß einnehmen wird.

Der potentielle Einsatz von Automatismen ist zumindest in folgenden Bereichen der Software-Erstellung generell "denkbar" (s. auch Abb.1):

a) Problemstellungs-Auswahl

b) Problemstellungs-Präzisierung

c) IA-Transformationsprozeß

d) Ausführung von A

Nun dazu einige Bemerkungen.

Zu d) Automatische Ausführung von Programmen (ein Artefakt-Teil) birgt keine grundsätzlichen Probleme und soll hier nicht näher diskutiert werden.

Zu c) Es werden derzeit internationale Workshops abgehalten, die sich mit der Thematik der Konstruktion allgemeingültiger, allumfassender, generischer Software-Prozeßmodelle befassen. Im Rahmen dieser Workshops liegt der Schwerpunkt in der Bereitstellung eines allumfassenden Meta-Modells, mit dessen Hilfe die Ableitung eines Bündels praktisch einsetzbarer Transformationsprozesse möglich wäre. Derzeit fehlt es allerdings an klaren Vorstellungen, wie dies eigentlich zu bewerkstelligen wäre. Zum Problembereich "Transformationsprozeß" folgen noch einige Überlegungen.

Zu b) Es gibt einige vage Vorstellungen, mit Hilfe von Automatismen in diesen Bereich vorzudringen. Oft tauchen diese Ansätze "irgendwo" auf der Ebene eines Transformationsprozesses auf. Ihre konzeptionelle Abgrenzung ist nicht ganz durchsichtig.

Zu a) "Vorläufig" nicht in Diskussion.

Problembereich "Transformationsprozeß"

Im folgenden soll auf die Interna des Transformationsprozesses (Modelle, Verhalten, Methoden, Techniken, Zwischenprodukte, usw.) nicht näher eingegangen werden.

Das Geschehen in diesem Bereich ist durch eine breitgestreute, theoretisch ausgerichtete Forschungsaktivität geprägt (s. oben), sowie durch das Bemühen, enger gefaßte, praktisch einsetzbare Ansätze, werkzeugunterstützt, möglichst schnell auf den Markt zu bringen. In der "Ermittlung" eines allumfassenden Meta-Modells, das die bedarfsweise Ableitung jedes erdenklichen Transformationsprozesses erlauben würde (durch Parametrisierung?), sehen manche das Problem der Software-Erstellung im wesentlichen, im besonderen dem "instrumentellen" Paradigma verdankend, schon bewältigt. An sich kann dieser Idee eine beachtliche Eleganz nicht abgesprochen werden. Sie ist aber nicht "problemfrei". Eines der fundamentalen Probleme liegt zum einen in der **tatsächlichen** Konstituierbarkeit eines derartigen Meta-Modells, zum anderen in der Findung eines begründbaren, durch stichhaltige Kriterien untermauerten optimalen (?) Weges von I nach A (Abb. 1). In der angestrebten universellen Anwendbarkeit dieses Ansatzes liegt der wesentliche "Problemmacher".

Problembereich "Instrumentarium"

Als Instrumentarium werden hier alle Hilfsmittel theoretischer (Methoden, Techniken, Verfahrensrahmen, usw.) und praktischer Art (Programme, sonstige Tools, usw.) verstanden.

Drei aktuelle Teilbereiche sollen hier kurz angerissen werden:

a) Marktsituation, Überbewertung der Wichtigkeit von Tools,

b) Vielfalt und Singularität von Tools, Tool-Integrierbarkeit, c) systematischer Einsatz des verfügbaren Instrumentariums.

Zu a) In der Praxis (besonders im Managementbereich) ist ein gewisser Hang zur Überbewertung der produktivitätssteigernden Eigenschaften von Tools bemerkbar. Diese Überbewertung basiert wahrscheinlich auf der Annahme einer weitestgehenden Algorithmisierbarkeit des SW-Erstellungsprozesses. Das Instrumentariums-Angebot erschöpft sich derzeit überwiegend in der Bereitstellung sogn. "universell einsetzbarer" CASE-Tools. Im Vordergrund steht dabei meistens eine toolmäßige Unterstützung "bester" Methoden bzw. Methodenkollektionen (lt. Prospektmaterial), die sich als ein Wundermittel zur Bekämpfung der Software-Krise eignen sollen. Für welche Kategorie von Problemstellungen das jeweilige "Tool" gut zu gebrauchen ist, geht meistens aus den Unterlagen nicht klar hervor.

Im übrigen sei hier die Anmerkung erlaubt, daß eine vernünftige Lösungssuche "traditionsgemäß" von einer gegebenen Problemstellung ausgeht, der sich erst dann eine Fragestellung nach einem allfälligen Lösungsinstrumentarium anschließt. Es ist kein Geheimnis, daß sich viele (u.U. nicht triviale) Problemstellungen auch klassisch *(Bleistift, Papier, Hirn)* effizient lösen lassen. Problemstellungen richten sich "selten" nach einem Lösungsinstrumentarium ! Hier die "übliche" Kausalkette der Lösungssuche:

Problemstellung → geeignetes Instrumentarium →
(geeignete Methodologie → geeignete Werkzeugunterstützung)

In diesem Zusammenhang drängt sich eine sehr spekulative Frage auf. Besteht nicht vielleicht die Möglichkeit, alle potentiellen, für die EDV-Realisierung praktisch sinnvollen, nichtpathologischen Problemstellungen so zu kategorisieren, daß jeder Kategorie, je nach Kategorie-Merkmalen, ein passendes Lösungsinstrumentarium zugeordnet werden könnte ?

Zu b) Der Software-Erstellungsprozeß umfaßt eine Fülle unterschiedlicher Teil-Tätigkeiten, für deren Bewältigung im Laufe der letzten 20 Jahre, je nach empfundener Dringlichkeit, eine reiche Anzahl unterschiedlicher Methoden und theoretischer "Rezepturen" entwickelt wurde. So ist ein Berg disparater methodischer Ansätze entstanden, wobei klarerweise jede Methode so ihre Vorzüge bietet. Was fehlt und erst recht im Laufe der geschichtlichen Entwicklung zunehmend als eine dringliche Notwendigkeit empfunden wird, ist die ganzheitliche Betrachtung des Entwicklungsprozesses. Diese Verschiebung in der Betrachtungsrelevanz, früher eine Methode im Mittelpunkt, heute eine Methodenkollektion, wirft die Frage auf, welche Methoden nun zu einer geeigneten Kollektion zusammenzustellen. Denn durch zwei wichtige Eigenschaften soll sich eine solche Zusammenstellung auszeichnen, nämlich

- durch eine möglichst einheitliche Anwender-Schnittstelle (Schnittstellenkonvergenz der Methoden) und
- eine "Nahtlosigkeit" der Methoden untereinander, die einen leichten Austausch von Zwischenergebnissen unter den Methoden erlaubt.

Der Methoden-Integrationsaspekt gibt derzeit ausreichend Anlaß, über weitere neue ganzheitliche Ansätze nachzudenken.

Zu c) Wie bereits ersichtlich, setzt ein SW-Erstellungsprozeß mehr als den Einsatz von Tools voraus. Bei einer auf Erfolg hin konzipierten Erstellung von Software sollten darüber hinaus mindestens folgende Punkte

besonders beachtet werden:

- fundierte **Kenntnis** der Fachdomäne (Problemstellungs-Spezifikation),
- **Auswahl** eines geeigneten theoretischen und praktischen Lösungsinstrumentariums (Kenntnis der Instrumentarien),
- **disziplinierter** Einsatz des gewählten Instrumentariums,
- **Motivation** der Teilnehmer des Erstellungsprozesses.

Freilich liegt das "Erfolgsgeheimnis" in einer ausgewogenen Mischung der angeführten Punkte, deren Zusammensetzung sich in Rezeptform kaum verordnen läßt. Die menschlichen Qualitäten und Fähigkeiten in ihren vielfältigen Erscheinungsformen nehmen dabei jedoch eine herausragende, kaum quantifizierbare Rolle ein.

Problembereich "Software als finales vs. evolutionäres Produkt"

Die steigende Komplexität neuer, in Frage kommender Problemstellungen der Gegenwart, zu deren Realisierung nicht zuletzt die Verfügbarkeit leistungsfähigerer Hardware "verleitet", zwingt über die bisherige Art und Weise der Software-Erstellung nachzudenken. Der bisherigen Praxis liegt überwiegend immer noch die Überlegung zugrunde, auf ein finales Produkt hin zu arbeiten (Dies betrifft bedauerlicherweise auch den Forschungsbereich !). Dieser Ansatz hat bei einfacheren Problemstellungen, deren Abstraktionen ausreichend "definiert" werden können, natürlich durchaus seine Gültigkeit (s. auch Abb.1).

Nun schlägt sich die Komplexitätszunahme natürlich nicht nur in den erstellten Produkten nieder, sondern auch in den Erstellungsprozessen, was in beiden genannten Fällen zu einer entsprechenden Aufwandssteigerung führt. Gerade dieser kontinuierlich wachsende Gesamtaufwand bringt es mit sich, daß über neue, zeitgemäße Ansätze nachgedacht wird. Insbesondere betroffen sind hier Problemstellungen mit einem ausgesprochenen Novitätscharakter, bei denen oft so gut wie keine Erfahrungswerte vorliegen (hohes "Unternehmensrisiko") und die erfahrungsgemäß mit beträchtlichen Investitionskosten verbunden sind.

In diesem Zusammenhang scheint eine dem Evolutionsprinzip Rechnung tragende Software-Erstellung der besser geeignete Verfahrensrahmen zu sein. Eine auf Evolutionsprinzipien basierende Software-Erstellung berücksichtigt in besonderer Weise die Tatsache, daß eine bedarfsorientierte, ökonomische Umsetzung komplexer Sachverhalte im Wege einer erkenntnisbedingten Systemevolution besser in den Griff zu bekommen ist. Deshalb erfreuen sich Lösungsinstrumentarien, die vom Konzept her das evolutive Vorgehen unterstützen, zunehmender Beliebtheit. Zu diesen Instrumentarien zählt u.a. das in vielfältiger Form anwendbare objektorientierte Paradigma (Anwendung bei der "Konstruktion" sowohl von Produkten, als auch von Prozessen), das derzeit zum Gegenstand intensiver Forschung und praktischer Erprobung gehört.

Einige besondere Vorzüge dieses Paradigmas, die im Rahmen einer evolutiven Software-Erstellung zum Tragen kommen (hier auf ein Produkt angewandt), mögen kurz angedeutet werden. (Inhaltliche Überschneidung der angeführten Punkte wird nicht ausgeschlossen).

Das objektorientierte Paradigma ...

- erlaubt dem Domänenfachmann eine problemorientierte ("objektorientierte") Sicht der Problemstellung,
- gestattet eine rasche, getreue und "problemlose" Überführung des "objektorientierten", vom Domänenfachmann spezifizierten Domänenmodells in das entsprechende Software-Systemmodell (architektonischer Entwurf),
- bietet ein "isomorphie"-bewahrendes *("Isomorphiebeziehung" zwischen Problemstellung, Architektur, Quellcode, Artefakt)* Präzisierungskonzept mit der Möglichkeit einer klaren konzeptuellen Trennung (und Berücksichtigung) von
- domänenspezifischen Sachverhalten ... I-Ebene,
- architektonischen Sachverhalten ... T-Ebene,
- implementationsspezifischen Sachverhalten ... A-Ebene,
- gestattet infolge der Isomorphie der Ebenen I, T und A untereinander einen "sprunglosen" Übergang von A nach T und I (Evolutionsschleife),

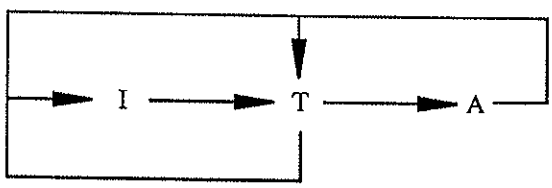


Abb. 2 evolutive Software-Erstellung, Skizze

- garantiert eine "konsistente" und aktuelle Dokumentation (als "Abfallprodukt").

Einer der wesentlichen Unterschiede zwischen einer finalen und einer evolutiven Software-Entwicklung besteht in der Bewältigung der Problematik der **"Softwarewartung"**, im finalen Ansatz der sogn. Wartungsphase.

Im finalen Ansatz der Software-Erstellung wird die Bedeutung der Wartungsproblematik trotz aller nachträglichen Korrekturen, nicht gebührend berücksichtigt. (Dieser Umstand hat vor allem entwicklungshistorische Wurzeln). Die "Rache" schlägt sich, wie die Erfahrung belegt, mit "harten" Zahlen zu Buche, denn

> Software-Einsatz, Gesamtaufwand (finaler Ansatz)
>
> =
>
> **1/3 Erstellung + 2/3 Wartung !**

Im evolutiven Ansatz hingegen ist die "Wartungsphase" (ein besserer Begriff wird wahrscheinlich auch hier noch gefunden werden müssen) besser in das gesamte Erstellungskonzept verteilt "eingearbeitet".

Nachsatz

Die derzeit spürbare Überbetonung des "Instrumentellen" sowie die Verharmlosung der Bedeutung des in vielfältigen Formen auftretenden **"Faktors Mensch"** kann sich später als eine "Wette auf das falsche Pferd" erweisen. Eine zielführende, bedarfsorientierte Software-Erstellung setzt primär eine **anthropozentrische** Ausrichtung voraus. Das nötige **Orientierungswissen** (im Gegensatz zum Verfügungswissen), an dem es derzeit scheinbar nicht nur hier mangelt, ist wahrscheinlich im Bereich der Wissenschafts- und Erkenntnistheorie zu suchen.

Literaturhinweise

Luft A.L. : Informatik als Technik-Wissenschaft, BI-Verlag, 1988

Meyer B.: Object-oriented Software Construction, Prentice Hall, 1988

Mittelstraß J. : Der Flug der Eule, von der Vernunft der Wissenschaft und der Aufgabe der Philosophie, suhrkamp 1989

Neuigkeiten von den Datennetzen im Bereich der Abt. Prozeßrechenanlage

Walter Kunft

1. ACONET, X.25-Netz für Österreichs Universitäten

Bereits im Frühjahr 1989 wurde vom ACONET-Verein dem Bundesministerium für Wissenschaft und Forschung ein Konzept für ein österreichweites X.25- Netz für Österreichs Universitäten übergeben. Dieses Konzept sieht einen Stufenplan vor, nach dem alle Universitäten durch Standverbindungen, die mit X.25 betrieben werden, zu einem Verbund zusammengeschlossen werden. X.25-Vermittlungsknoten in sämtlichen Universitätsstädten sind miteinander in einer Ring-Topologie verbunden, um Alternativwege sicherzustellen. An diese Knoten, die Standortknoten oder S-Knoten genannt werden, werden in den einzelnen Universitätsstädten Endknoten (E-Knoten) der einzelnen Universitäten angeschlossen. S- und E-Knoten bieten außerdem einen Zugang zu DATEX-P, dem öffentlichen Datenpaketvermittlungsnetz der ÖPTV. Der Stufenplan sieht für die Verbindung der Vermittlungsknoten zunächst eine Übertragungsgeschwindigkeit von 9600 bps (DDL 9600), später eine von 64 kbps (DS 64) vor.

Die Basis von X.25 ermöglicht den Einsatz von herstellerspezifischen Kommunikationsarchitekturen wie SNA und DECnet aber auch von herstellerunabhängigen Kommunikationsprotokollen, wie TCP/IP (mit allen zugehörigen Anwendungsprotokollen), X.400 und OSI. Das Netz stellt somit die Basis für das seit Jahren geplante Akademische Computernetz ACONET dar, das einen herstellerunabhängigen Verbund mit OSI-Protokollen zum Ziel hat. Das X.25-Netz wird es erstmals ermöglichen, daß Universitäten in ganz Österreich ungehindert miteinander kommunizieren und mit gleichen Möglichkeiten an folgenden internationalen Datennetzen teilnehmen können: EARN, EUnet, X.400, HEPNET, IXI (International X.25 Infrastructure, ein europäischer akademischer Verbund, der im Rahmen von COSINE errichtet wird) und später auch INTERNET.

Im Herbst 1989 konnte vom BMWF die Finanzierung einer ersten Stufe im Jahr 1990 zugesagt werden. In dieser ersten Realisierungsstufe werden Vermittlungsknoten in allen Universitätsstädten durch DDL-9600-Verbindungen ringförmig zusammengeschlossen. Die Verbindung der Universität Wien mit der Universität Linz wird bereits in dieser ersten Stufe mit 64 kbps (DS64) realisiert. Die Vermittlungsknoten enthalten zunächst eine Minimalausstattung im Rahmen der finanziellen Möglichkeiten, sind jedoch kapazitäts- und geschwindigkeitsmäßig flexibel ausbaubar, sodaß Erweiterungen jederzeit möglich sind. In dieser Stufe werden in folgenden Universitäten Vermittlungsknoten mit unterschiedlicher Leistungsfähigkeit eingesetzt:

Uni-Wien	S-Knoten
TU-Wien	S-Knoten
WU-Wien	E-Knoten
TU-Graz	S-Knoten
Uni-Graz	E-Knoten
Uni-Leoben	S-Knoten, zunächst mit der Leistungsfähigkeit eines E-Knotens
Uni-Klagenfurt	S-Knoten, zunächst mit der Leistungsfähigkeit eines E-Knotens
Uni-Innsbruck	S-Knoten
Uni-Salzburg	S-Knoten, zunächst mit der Leistungsfähigkeit eines E-Knotens
Uni-Linz	S-Knoten

Zur Durchführung eines zentralen Netzmanagements wird an der **TU**-Wien ein Netzkontrollzentrum aufgestellt.

Die S-Knoten sind in einer Ringstruktur miteinander gekoppelt, E-Knoten werden an die S-Knoten sternförmig angeschlossen. In Wien werden zwei S-Knoten errichtet, da sowohl von der Universität Wien als auch von der **TU**-Wien internationale Verbindungen ausgehen, die alle anderen Universitäten möglichst effizient nutzen können sollen.

Zur Beschaffung der Vermittlungsknoten wurde Anfang Februar 1990 vom Bundesministerium für Wissenschaft und Forschung eine öffentliche Ausschreibung durchgeführt. Die Ausschreibungsunterlagen wurden von der Abt. Prozeßrechenanlage in Koordination mit dem ACONET-Verein vorbereitet. Bei der Anbotseröffnung am 21.3.1990 haben vier Firmen Anbote im vorgesehenen Preisrahmen abgegeben. Diese werden derzeit von der Abt. Prozeßrechenanlage beurteilt.

Parallel dazu wurden die erforderlichen DDL-9600-Verbindungen und die DS64- Verbindung zur Realisierung des Ringes bei der ÖPTV beantragt. Außerdem werden Gespräche mit der ÖPTV geführt, um ein möglichst effizientes Adresskonzept zu erarbeiten, mit dem ACONET in DATEX-P eingebunden werden kann.

Wenn keine unvorhergesehenen Hindernisse auftreten, werden die Lieferung und die Abnahme der Geräte des Bestbieters sowie die Aufnahme des Probebetriebes des neuen X.25-Netzes für Österreichs Universitäten im 3. Quartal 1990 durchgeführt werden können.

2. EUnet, European UNIX network

Die Teilnehmerzahlen am österreichischen EUnet-Hauptknoten, namens "tuvie", sind auch im Jahre 1989 konstant angestiegen. Der Verkehr, vor allem bei den News, ist so stark angestiegen, daß die Anmietung einer Standleitung nach Amsterdam geplant ist, da DATEX-P bereits zu teuer geworden ist.

Im Jänner 1990 wurden auf "tuvie" neue Betriebssystemversionen (ULTRIX, V3.0 und kurz danach V3.1) installiert. Damit verbunden war auch die Installation eines neuen ULTRIX-DECnet-Mailers, der es nun ermöglicht, electronic mail für nicht lokal erreichbare Rechner und Netze außerhalb von EUnet an ein Gateway-System an der Universität Wien, namens "auniw", weiterzuleiten. Damit ist es nun möglich, daß Universitätsinstitutionen aus dem österreichischen Teil von EUnet e-mail direkt und kostenlos ins BITNET/EARN senden können.

3. IXI, International X.25 Infrastructure

Im Rahmen des EUREKA-Projektes COSINE (Cooperation for OSI Networking in Europe) wird seit Beginn des Jahres 1990 ein Pilotverbund, namens IXI (International X.25 Infrastructure), errichtet, an dem, dank der laufenden und intensiven Mitarbeit des ACONET-Vorstandes in RARE, auch Österreich teilnimmt. Abb. 1 zeigt die Topologie der ersten Phase von IXI.

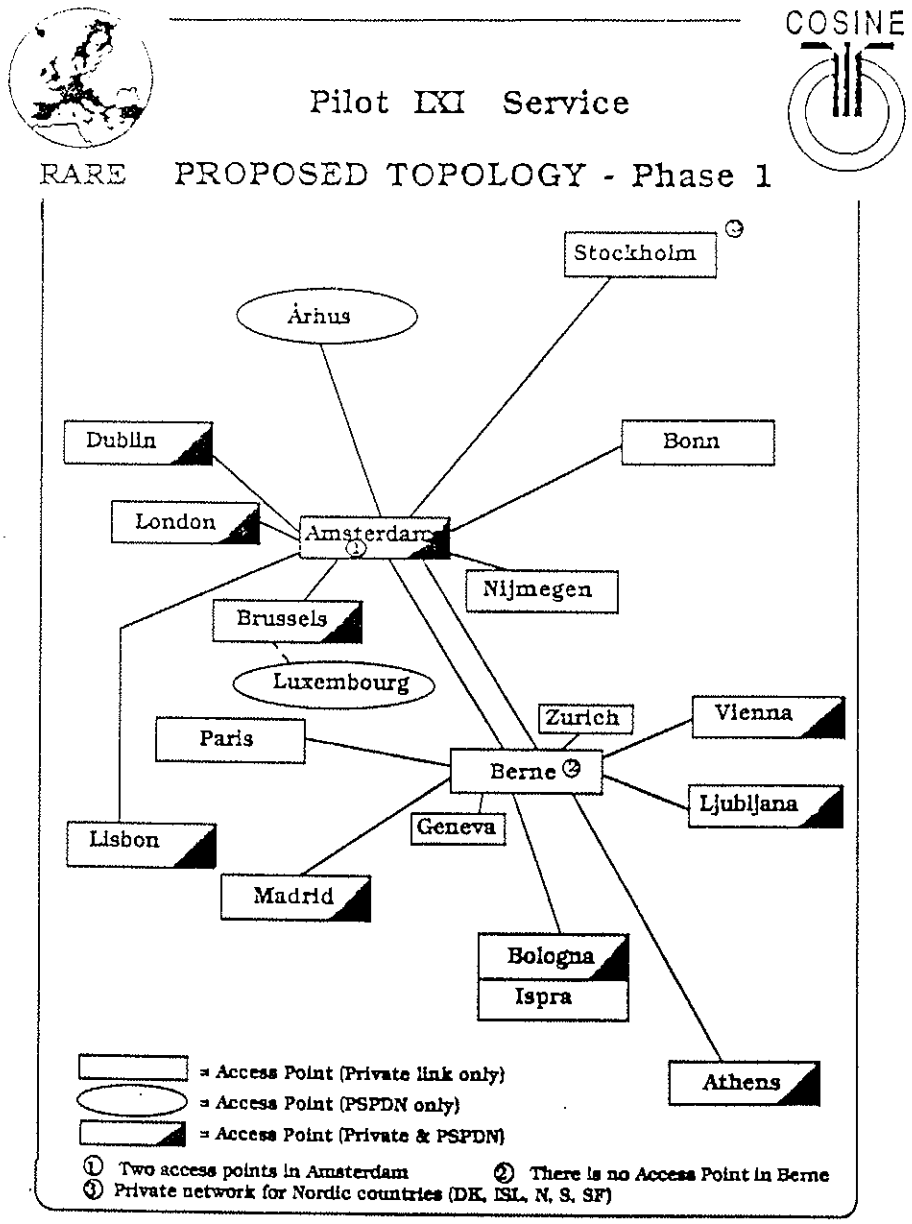


Abb.1: IXI Topologie

Seit Ende Jänner 1990 ist der X.25-Knoten der Abt. Prozeßrechenanlage, der zu diesem Zweck ausgebaut wurde, mit 64 kbps am IXI-Knoten Wien angeschlossen. Wir waren damit eine der ersten Universitäten in Europa (!), die an diesem neuen Verbund teilnahmen. Über den Knoten der PRA kann im ersten Halbjahr 1990 neben der **TU**-Wien auch das IEZ und die Universität Wien, die beide mit unserem Netzknoten über eine Standleitung gekoppelt sind, an IXI herangeführt werden. Im zweiten Halbjahr 1990, wenn ACONET errichtet wird, wird an Stelle unseres derzeitigen X.25-Knotens ein leistungsfähiger ACONET-Knoten an IXI gekoppelt werden, der allen Universitäten, die an ACONET angeschlossen sind, an IXI heranführen wird.

Die Benützung von IXI ist während der Dauer der Pilotphase (1 bis 3 Jahre) kostenlos. Der neue Verbund wird für die **TU**-Wien und alle anderen österreichischen Institutionen eine Möglichkeit bieten, mit akademischen Institutionen anderer Länder zu kommunizieren und Netze zu erreichen, mit denen bisher noch nicht auf direktem Wege Daten ausgetauscht werden konnten (z.B. SURFNET). EARN hat bereits seine Absicht erklärt, IXI in Zukunft verwenden zu wollen. Von EUnet steht eine solche Erklärung noch aus. Auch HEPNET wird für Österreich über IXI erreichbar sein.

4. **TU**NET wurde X.25-fähig

Bereits vor langer Zeit planten wir den Anschluß von **TU**NET an Wide-Area-Networks über X.25. Zu diesem Zweck wurde für den CISCO-Router am Freihaus Ende 1989 X.25-Software beschafft. Diese Software ermöglicht TCP/IP-Verbindungen aus dem gesamten **TU**NET zu anderen TCP/IP-Netzen im nationalen und internationalen Bereich. Der CISCO-Router fungiert dabei als Gateway, der zu Gateways anderer Bereiche, die ebenfalls TCP/IP über X.25 abwickeln können, X.25- Verbindungen aufbaut und darüber dann TCP-Verbindungen legt. Auf diese Weise kann jeder TCP/IP-fähige **TU**NET-Teilnehmer mit überregionalen TCP/IP- Netzen Verbindung aufnehmen.

Die Software wurde Anfang 1990 im CISCO-Router in der Wiedner Hauptstraße installiert. Damit die X.25-Funktionalität möglichst optimal ausgenützt werden kann, wurde der CISCO-Router mit dem X.25-Switch der PRA verbunden (19,2 kbps). Damit können alle überregionalen X.25-Verbindungen, die von diesem Switch ausgehen, von allen **TU**NET-Teilnehmern verwendet werden. Interessant sind dabei vor allem Standverbindungen, da TCP/IP über DATEX-P zwar möglich aber teuer ist. Die derzeit von unserem Switch weggehenden Standverbindungen, die in diesem Zusammenhang von Interesse sind, führen zur Universität Wien und zu IXI. Eine weitere Standverbindung nach Amsterdam ist geplant. Über sie wird der gesamte EUnet-Newsverkehr und ein Großteil des EUnet Mail-Verkehrs unter Einsatz von TCP/IP abgewickelt werden. Sie könnte den Bereich der **TU**-Wien auch ans INTERNET führen.

Ende Februar 1990 wurden gemeinsam mit der Abt. Digitalrechenanlage die ersten Tests durchgeführt. Dabei konnte eine TCP/IP-Verbindung erfolgreich vom CISCO-Router über den X.25-Switch der PRA und über DATEX-P zum IEZ und später auch zum Rechner "mcsun", dem europäischen EUnet-Hauptknoten in Amsterdam, erstellt werden. Tests mit Partnern, die über IXI erreichbar sind, sind geplant.

Durch diesen neuen Dienst, der sich derzeit noch im Teststadium befindet, werden die Kommunikationsmöglichkeiten aus dem gesamten Bereich der **TU**-Wien in alle Welt wesentlich verbessert werden können.

Maildienste

Martin Rathmayer

Seit der Errichtung des **TU**NET hat der Anschluß von Netzteilnehmern rasant zugenommen. Besonders der preisgünstige Anschluß von PC's hat die Möglichkeit geschaffen, **TU**NET für den schnellen und problemlosen Einsatz als elektronisches Post-Medium zu verwenden. Damit nun dieser ePost-Verkehr sowohl innerhalb des **TU**NET, als auch in externe Netze jederzeit gewährleistet ist, stellt das EDV-Zentrum diverse HW- und SW-Unterstützungen zur Verfügung.

Auf der untersten Ebene sorgen Router, Bridges, Repeater, Datex-P Anschlüsse, Modem's, usw. für den Zusammenschluß von Netzsegmenten und den Anschluß an externe Netze. In höheren Ebenen fungieren EDV-Zentrums-Rechner als Backbones, Gateways, Protokollkonverter, Nameserver, usw.. Mit diesem Mailverbund ist es nun, bis auf einige wenige Ausnahmen, möglich zwischen zwei am **TU**NET - oder vom **TU**NET aus erreichbaren Netzen - angeschlossenen Rechnern eMail's auszutauschen. Dazu ist nur die Kenntnis vom Umgang mit dem Mailer am eigenen Computer und einige Informationen über protokoll- und netzspezifische Adreßsyntaxen notwendig. Diese lästige Auseinandersetzung mit Netzadressierungsarten bleibt dem Benutzer momentan nicht erspart, wird aber mit zunehmender Vereinheitlichung, internationaler Normung und Einsatz von intelligenteren Mailservern immer geringer werden.

Um dem Netzneuling das Versenden von ePost etwas zu erleichtern, hat das EDV-Zentrum ein ePA-Verzeichnis herausgebracht, das sowohl als Papierform als Anhang zum Telefonverzeichnis versendet wird, als auch **online** auf dem Rechnerknoten `EGH780` zur Verfügung steht. Dieses Verzeichnis enthält außer einer Benützungsanleitung als wesentlichen Bestandteil das eigentliche Teilnehmerverzeichnis. Hier sind Personen, die per eMail im **TU**NET erreichbar sind, auf eigenem Wunsch mit allen für eine eindeutige Adressierbarkeit nötigen Daten eingetragen. Das sind Lokaladresse, Knotenname, Netzzugehörigkeit, Protokoll, usw. Möchte ein Benutzer nun von seinem Computer aus eine eMail an eine im Verzeichnis registrierte Person schicken, muß er lediglich aus allen angegebenen Daten laut Benützungsanleitung die Zieladresse bilden. Würde zum Beispiel ein Benutzer der `AWITUW01` eine eMail an mich schicken, müßte er in seinem Mailer als Adresse

```
RATHMAYER@EGH780.TUWIEN.AC.AT
```

eintippen. Das Verzeichnis wird online laufend auf den neuesten Stand gebracht und erscheint in Papierform jeweils am Anfang eines Wintersemesters.

Die Möglichkeiten der ePost beschränken sich aber nicht nur auf den Austausch von Botschaften. Durch den Anschluß an externe Netze wie USENET, BITNET, usw. kommt der Benutzer in den Genuß der Verwendung von Info- und Listservern sowie der Teilnahme an Konferenzsystemen. Als Beispiel seien die

Listserver TRICKLE (Simtel) und RICE im BITNET und der

NEWS-Dienst im USENET genannt.

Am **TU**NET selbst werden unter anderem ein

Infoserver und ein

VAX-NOTES-Konferenzsystem angeboten.

Trotz eines relativ großen administrativen und technischen Verwaltungsaufwandes ist die Benützung von speziellen Maildiensten nur mit geringen Kosten verbunden.

Für detaillierte Auskünfte stehe ich gerne zur Verfügung:

Telefon Kl. 5502,

ePost: `egh780::rathmayer` oder
`rath@tuvie.`

Der Scanner, ein Werkzeug für Desktop Publishing

Robert Meixner

An der Abt. Prozeßrechenanlage steht ein Flachbettscanner der Firma MICROTEK vom Typ MSF-300G zur Verfügung. Die Maximalauflösung dieses Scanners beträgt 300 dpi (11.8 dots/mm) horizontal und vertikal. Er kann in 3 verschiedenen Scanmodi betrieben werden. Der 'line art' Mode dient zum scannen reiner S/W Vorlagen, im 'halftone mode' können auch Graustufen durch das Zusammenfassen mehrerer Bildpunkte erfasst werden, wodurch natürlich die Auflösung reduziert wird. Der 'grayscan' Mode ermöglicht es, jeden Bildpunkt mit bis zu 256 Graustufen abzutasten. Abgesehen von den Scanmodi kann auch noch der Kontrast, die Helligkeit und die gewünschte Auflösung eingestellt werden. Die maximale Dokumentengröße beträgt 216mm x 356mm. Durch die Flachbettausführung ist es auch problemlos möglich, Bücher als Original zu verwenden. Für den Scanner stehen drei Softwarepakete zur Verfügung: RECOGNITA ist ein Softwarepaket, welches das optische Einlesen von maschinengeschriebenen und gedruckten Texten erlaubt. Somit entfällt das lästige Neuschreiben bereits vorhandenen Textmaterials. Nicht nur Originaltexte, sondern auch Fotokopien sind verwendbar, allerdings ist die Fehlerquote stark von der Qualität des Originals abhängig, und steigt bei schlechten Vorlagen dementsprechend an. Die eingelesenen Texte werden im ASCII-Format abgespeichert, und können somit mit den meisten Editoren und Desktop Publishing Programmen weiter bearbeitet werden. EYESTAR PLUS ermöglicht es, Grafiken mit einer Auflösung von 300 dpi im 'line art' mode und 'halftone' mode zu scannen, zu bearbeiten, und in verschiedenen Fileformaten auszugeben. Die zur Verfügung stehenden Formate EyeStar, Windows Paint, TIFF und PC Paintbrush können mit den meisten Desktop Publishing Programmen, wie z.B. Ventura Publisher und Pagemaker, gelesen werden. GRAYSCAN erlaubt es Grafiken mit einer Auflösung von 300 dpi mit je 256 Graustufen pro Bildpunkt zu scannen. Die Ausgabe der Files erfolgt im TIFF Format.

Die dargestellten Abbildungen wurden mit dem Programm EYESTAR gescannt. Als Vorlage diente ein SW-Foto. Abbildung 1 wurde im 'line art' Mode abgetastet, d.h. alle Graustufen gehen verloren. Alle Bildpunkte die ein bestimmtes Helligkeitsniveau überschreiten werden weiß, alle anderen schwarz dargestellt. Abbildung 2 hingegen wurde im 'halftone' mode gescannt. Es werden jeweils 8*8 Bildpunkte zur Darstellung einer Graustufe zusammengefaßt. Dadurch können nun auch Bilder die mehrere Graustufen beinhalten, dargestellt werden. Es tritt aber ein Verlust an Auflösung und Kontrast auf. Bilder im 'grayscan' Mode können aus drucktechnischen Gründen leider nicht dargestellt werden, da mit dem verwendeten Laserdrucker nur eine Auflösung von 300 dpi erreicht wird. Und auch bei den dargestellten Bildern tritt durch den XEROX Kopiervorgang bei der Reproduktion ein starker Qualitätsverlust auf.

Interessenten können sich an Herrn Meixner Robert (Klappe 3689) wenden. Der Aufstellungsort des Scanners ist das Neue Elektrotechnische Institut Gußhausstr. 27-29 5. Stock.

Abbildung 1

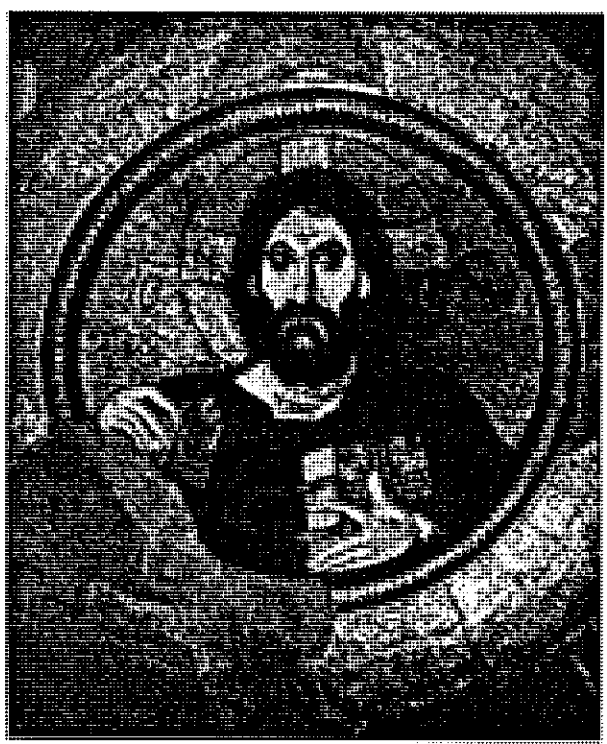


Abbildung 2

Unterstützung der Institutsgebarung: "IGe"

Georg Gollmann

Jedes Institut benötigt ein gewisses Maß an Buchhaltung, sei es zur Verfolgung der Mittel, die - im Rahmen der ordentlichen und außerordentlichen Dotation - über die Quästur laufen, sei es zur Abrechnung von Projekten mit der Wirtschaft.

Als im Laufe des Jahres 1988 die Universitätsinstitute beschränkte Rechtsfähigkeit bezüglich Drittmittelprojekten erhielten und von ihnen eine entsprechende Buchhaltung verlangt wurde, entwickelte die Abt. Prozeßrechenanlage ein bis dahin intern verwendetes Programm zur Abdeckung des neuen Bedarfes weiter. Entsprechend den Benutzeranforderungen wurde das bisher rund 40 mal verteilte Programm mehrmals verbessert und erweitert. Nunmehr steht das Programm sowohl unter dem Betriebssystem MS-DOS als auch unter VAX/VMS zur Verfügung. Darüberhinaus kann es jetzt neben der Abrechnung von Drittmittelprojekten auch zur Kontrolle der über die Quästur laufenden Zahlungen dienen.

Einige Stichworte zum Leistungsumfang der Version 2.0:

- 2000 Buchungszeilen
- 200 Konten und Verrechnungskreise
- Beträge bis 999.999.999,99
- Zuordnung der Buchungen zu Verrechnungskreisen, Möglichkeit der getrennten Abrechnung nach einzelnen Kreisen oder vordefinierten Gruppen von Verrechnungskreisen.
- Zusammenfassung von Konten in Sammelkonten, sodaß Kontengruppen gemeinsam abgerechnet werden können. Sammelknten können ihrerseits in ein übergeordnetes Sammelkontoeingebracht werden.
- Aufteilen eines Geschäftsfalles auf mehrere Buchungszeilen (mit der gleichen Buchungsnummer), wenn Teilbeträge auf verschiedene Konten gebucht werden sollen.
- Belegung der Buchung mit einer Buchungsphase (offen, freigegeben, erledigt, storniert, ungültig), Berücksichtigung der Phasen in der Kontenabrechnung.
- Verknüpfung von Bestellungen und Rechnungen, sodaß Teillieferungen in der Abrechnung korrekt berücksichtigt werden. Gleiches gilt für Auftragsbestätigungen und Zahlungseingänge.
- Paßwortschutz für die Datenbestände
- Übertragsprogramm: Wahlweise Summen- oder Saldenübertrag zum Zusammenfassen erledigter Buchungen in einer Übertragsbuchung je Konto und Verrechnungskreis.
- Benutzerführung über Menüs und Bildschirmmasken
- Gedruckte Dokumentation und Beispieldatensatz
- Datensätze aus früheren Programmversionen können mit einem Konvertierungsprogramm übernommen werden.
- O.Dot. Version: Beträge und Konten können nachträglich an dietatsächlichen Quästurbuchungen angeglichen werden.

Das Programm wird an Institute der **TU**-Wien kostenlos weitergegeben, von anderen Beziehern wird ein Kostenersatz von S 2000,- eingehoben.

Nähere Auskünfte sind beim Autor unter der Klappe 3611 oder den

ePost-Adressen `EGH780::GOLLMANN` oder

`gollman@tuvie` zu erhalten.

Softwareangebot der PRA

Georg Gollmann, Günter Houdek, Jadwiga Rogl, Rudolf Sedlaczek, Günther Wehrberger

Dieser Artikel bietet einen Überblick über die an der Abt. Prozeßrechenanlage vorhandene Software. Die Programme sind in drei Kategorien eingeteilt:

- Kommerzielle Produkte, über die Erfahrungsberichte aus erster Hand verfügbar sind
- Programme, die an den Rechenanlagen der PRA angeboten werden
- Programme, die aufgrund von Campuslizenzen oder die als Public Domain bzw. Shareware Programme an Institute der **TU**-Wien weitergegeben werden können.

Die Aufstellung ist nach Systemen gegliedert.

1.0 Apollo (Rogl)

Auf den 4 Apollo's an der PRA (DN4500, DN3500, DN3010, DN3000), die über Domain-Ring miteinander verbunden sind, sind folgende SW-Produkte verwendbar:

- Betriebssystem OS 9.7.5
- Domain/IX (Unix fuer OS 9.7.5)
- Betriebssystem Domain OS 10.1
- X11-Windows fuer 10.1
- DOS-Emulator fuer 9.7.5 und 10.1
- X11 Release 4 fuer 10.1 (10.2)
- Visula 4.2 unter OS 9.7.5
- Informix SQL-Rel. Datenbanksystem unter OS 9.7.5

2.0 Macintosh (Gollmann)

2.1 Kommerzielle Produkte

Excel 1.5	Tabellenkalkulation
PageMaker 3.0	Layout
Fontographer 2.4.1	Erstellung von Postscript Schriften
Fontatstic 2.8	Erstellung von Bitmap Schriften
FreeHand 2.0	Grafik (PostScript)
CricketDraw	Grafik (PostScript)
Smalltalk-80 2.3	Smalltalkimplementation von ParcPlace Systems
Smalltalk/V Mac 1.1	Smalltalkimplementation von Digitalk
MacMETH 2.5	Modula-2 Entwicklungsumgebung

2.2 PD & Shareware

2.2.1 Virenbekämpfung

Das leidige Virenproblem ist durch, meist nicht sehr exakte Presseberichte, allgemein bekannt. Durch entsprechende Vorsichtsmaßnahmen läßt es sich aber durchaus in den Griff bekommen. Am wichtigsten sind wohl organisatorische Maßnahmen, das heißt etwa, daß keine Disketten oder Programme unbekannter Herkunft (ohne gründliche Kontrolle) verwendet werden. Quellen wie USENET und Simtel haben sich bisher als virenfrei erwiesen. Zur Verhinderung eines Virenbefalls, bzw. zur Entfernung von Viren sind eine Reihe von Hilfsprogrammen erhältlich. Wichtig ist, immer die neuesten Versionen dieser Werkzeuge einzusetzen, um auch alle bekannten Viren zu erfassen. Die PRA ist bemüht, ihr Angebot stets am neuesten Stand zu halten. Aus der unten angeführten Liste wären besonders *GateKeeper* samt *GateKeeper Aid* zur Vermeidung von Infektionen, sowie *VirusDetective* zur Entdeckung und Beseitigung von Virenbefall, hervorzuheben.

AntiPan 1.0	Disinfectant 1.5	GateKeeper 1.1.1
GateKeeper Aid 1.0.1	Interferon 3.1	RWatcher 1.0
Vaccination	Vaccine 1.0.1	Virus RX 1.6
VirusBlockade™ II Ltd. 1.0	VirusDetective 3.1	VirusWarning INIT
Warning		

2.2.2 Kommunikation

- NCSA Telnet	Telnet (VT100 und Tektronix 4014 Emulation) und FTP (Filetransfer) über Ethernet bzw. LocalTalk Verkabelung.
- tn3270	TN3270 Emulation über Ethernet bzw. LocalTalk
- Kermit	Terminalemulation und Filetransfer über serielle Leitung
- Public Folder	Filetransfer zwischen Mac´s über LocalTalk
- Janus	Einfaches Drucker-Gateway: Kopiert von der seriellen Schnittstelle zu einem Laserdrucker am LokalTalk.
- ChainMail	MailServer für AppleShare Fileserver

2.2.3 HyperCard

Zum Standardprogramm HyperCard existieren eine Reihe Stacks, die nützliche Scripts, externe Befehle (XCMD, XFNC) Grafiken und Sounds enthalten. Eine Auswahl der an der PRA verfügbaren "Goodies:"

- Clipboard	Kopieren von der / in die Zwischenablage in einem Script
- CommandHistory	Speichert die in die MessageBox eingegebenen Kommandos
- Inkey	Lesen einzelner Tastendrücke
- Resource Library	Sammlung diverser XCMDs, XFNCs, Sounds etc.
- Home Improvements	Verbesserungen für den Home Stack
- HyperEdit	Editor für Scripts
- HyperZap	Löscht den Passwortschutz von Stacks
- PopUpMenu	PopUp Menüs
- ScrollControl	Scrollbar als Button

2.2.4 Technical Notes

- Technical Notes	Eine Sammlung von mittlerweile über 260 Artikeln zu verschiedensten Hardware und Software Fragen am Mac. Herausgegeben von Macintosh Developer Technical Support (Mac DTS) als MacWrite Files.
- TN Stack 3.0	Technical Notes als HyperCard Stack.
- Human Interface Notes	Diese Notes geben Hinweise zur Gestaltung der Benutzerschnittstelle von Programmen. Die Information ist auch für NICHT-Macintosh Umgebungen von Nutzen.

2.2.5 Archivierung

- Stuffit	Komprimieren von Files und ganzen Directory Bäumen. Liest und schreibt auch BinHex 4 Format. BinHex/Stuffit ist das gängigste Format in Archiven und Newsgroups für Mac-Files.
- ArcMac.Arc	Liest das PC-ARC Format
- MacBooz 2.0	Liest das PC-ARC Format
- tar v2.0	Liest und schreibt UNIX tar files
- MacCompress	Liest und schreibt UNIX compress files

2.2.6 Grafik / PostScript

- DistillPS	Wandelt Postscriptcode so um, daß er schneller druckt.
- Error Handler	Debug-Utility für Postscriptprogramme am Laserdrucker
- MandelZot	Mandelbrot Grafik
- Image	Paintprogramm (nur für 8-bit Videokarten)
- VisionLab	Darstellung von GIF Bildern
- Klutz	Zum Speichern, Setzen und Manipulieren von CLUTs

2.2.7 Textverarbeitung / Layout

- OzTeX 1.2	TeX Implementation für den Mac
- FaceLift	Schriftattributänderung in MacWrite Files
- MakeWrite	Wandelt ASCII Files mit Formatmarken in MacWrite Files um

2.2.8 Hilfsprogramme

- Variants	Übersetzung zwischen verschiedenen Zeichensätzen (Mac, PC, DEC)
- XForm	Bit-, byte- und zeilenweise Transformation von Files
- Moire	Screensaver
- SCSI Evaluator	Testet den Durchsatz von SCSI Platten
- SF&I	Utility zum Formatieren und Installieren von SCSI Platten
- SFScrollInit	Standard-File-Dialoge merken sich die Position in der File-Liste
- MacSnoop	Filecditor (HEX)
- MacBinary	Wandelt vom / ins MacBinary Format um
- AppMenu3.4	Installiert Menü der aktiven Applikationen
- NumberCrunch	Wissenschaftlicher Rechner mit X/Y Diagramm
- ClipboardMagician	Deskaccessory zur Manipulation des Clipboardinhaltes, wie etwa die Umwandlung zwischen verschiedenen Datenformaten; erweiterbar.

Eine ausführliche Liste des aktuellen Angebotes ist beim Autor erhältlich. Darüberhinaus kann Unterstützung bei der Beschaffung nicht aufgelisteter Software gegeben werden.

3.0 MS-DOS (Wehrberger)

Die in diesem Abschnitt beschriebenen Programme stehen zur Benützung im Graphik- Arbeitsraum der PRA (Gußhausstraße, 5.Stock) zur Verfügung.

3.1 AUTOCAD V10

Allgemeines CAD-Programm für unterschiedliche Fachgebiete. 2.5D- und 3D-Datenmodell. 3D als Kanten- und Flächenmodell. Dateneingabe grundrißorientiert, aber auch durch vordefinierte Körper und durch freie dreidimensionale Beschreibung. Schnittstellen: IGES, DXF. Ausgabeformate: HPGL, Postscript. Gleichzeitige Darstellung mehrerer Ansichten am Bildschirm.

3.2 AUTOSHADE

Schattierung von AUTOCAD-Zeichnungen mit variabler Lichtquelle und Ausblendung verdeckter Kanten.

3.3 POINT-LINE

CAD-Programm für Entwurf, Konstruktion und Design für bauspezifische Anwendungen. 2-dimensionales und 3D-Datenmodell (Kanten- und Flächenmodell). Dateneingabe grundriß-, ansicht-, und schnittorientiert, sowie durch vordefinierte Körper. DXF-Schnittstelle. Datenausgabeformate: IGES, HPGL und Postscript. Gleichzeitige Darstellung von Grund-, Auf- und Seitenriß, sowie einer beliebigen Perspektive. Darstellung der Perspektive als Festkörper mit Definition von Lichtquellen, Schatten und Durchsichtigkeit. Innenraumperspektiven, Fly-Through und Walk-Through-Funktionen sowie die farbliche Weiterverarbeitung von 2D- oder 3D-Entwürfen sind weitere interessante Eigenschaften für den Architekten.

3.4 SPIRIT

CAD-Programm für den Architektur- und Baubereich. 2.5D- und 3D-Datenmodell (Kanten- und Flächenmodell) mit einheitlicher Datenbasis. Dateneingabe grundriß-, ansicht- und schnittorientiert, sowie durch vordefinierte Körper und durch freie dreidimensionale Beschreibung. DXF-Schnittstelle. Datenausgabeformat: HPGL. Eine integrierte Datenbank ermöglicht die Zuordnung von Material- und Konstruktionseigenschaften zu Zeichnungselementen. Darstellung beliebiger Parallel- und Zentralprojektionen, Innenraumperspektiven, Fly-Through und Walk-Through-Funktionen. Ein breites Spektrum architekturspezifischer Zusatzfunktionen erleichtert die Arbeit des Architekten.

4.0 SUN (Houdek)

Die Sun/Sparc 330 an der Abt. Prozeßrechenanlage dient als Nachfolge des ausgedienten Calma-Systems (Eclipse mit GDS II). Die Hauptanwendungssoftware ist das Softwarepaket **EDS III**, das Nachfolgepaket der GDS II-Software. Daten vom alten GDS II-System können zur neuen EDS III-Software übernommen werden, wenn diese im Stream-Format (GDS II Befehl OUTFORM) zur Verfügung stehen.

Die Systemumgebung umfaßt:

- UNIX BSD4.2
- TCP/IP
- NFS
- SunView
- X-OpenWindows (X11/NeWS, kombinierter Server)
- X11R4 (original MIT)
- C-Compiler

In Kürze wird die nächste Version von EDSIII auf der SUN installiert, das dann den Namen CONSTRUCT tragen wird. Dieses Paket ist ein neues Produkt der Firma VALID, welches das GDSII (EDSIII) sowie den von VALID entwickelten Layouteditor LED beinhaltet. Ausgabe auf Penplotter (HP), Photoplotter (Gerber) und Patterngenerator werden unterstützt. Nähere Informationen bezüglich EDSIII bzw. CONSTRUCT sind im Artikel über die Calma Nachfolge zu finden.

5.0 VAX/VMS (Sedlaczek)

5.1 Produkte, die an den Anlagen der PRA zur Verfügung stehen

5.1.1 Betriebssystem

VMS	inkludiert DECwindows (X Window System) und VAXcluster	
DECnet		
VWS Window System		
DFS	Distributed File Service	(im Testbetrieb)
DNS	Distributed Name Service	(im Testbetrieb)
DQS	Distributed Queue Service	(im Testbetrieb)

5.1.2 Compiler

VAX-Fortran

Pascal

C

5.1.3 Programmentwicklungswerkzeuge

VAXSET, enthält

LSE	Language Sensitive Editor
SCA	Static Source Code Analyzer
PCA	Performance and Coverage Analyzer
CMS	Code Management System
MMS	Module Management System
DTM	DEC Test Manager

5.1.4 Netzwerk-Software

P.S.I. X.25 Software für DECnet und Mail

PCSA Personal Computer Systems Architecture, umfaßt:

VAX/VMS Services für MS-DOS (File-, Disk- und Print-Service)

PCSA Client, inkludiert DECnet-DOS und MS-Windows

CMU-TEK TCP/IP Implementation der Carnegie-Mellon University

UCX Ultrix-Connection, umfaßt NFS und FTP (im Testbetrieb)

5.1.5 Sonstige Produkte

Notes	Konferenzsystem
MAPLE	Symbolische Mathematik
SEDT	Screen EDiTor, EDT kompatibler Editor für VMS, MS-DOS, OS/2, Ultrix, Ultrix-Risc und SCO-Xenix
NAG Library	Mathematische und statistische Unterprogramme
EUCLID-IS	3D-Volumenmodell für mechanische Konstruktion
UNIRAS	Unterprogramm-Bibliothek für Raster- und Vektorgraphik
UNIRAS Uniedit	Graphischer Editor
SIGHT	Graphischer Editor für VAXstations unter VWS

5.1.6 Nicht unter Wartung stehende Produkte

GOSY	Graphical Output System, graphische Unterprogrammbibliothek
EAN	X.400 Mail

5.2 Produkte, die an Institute der TU-Wien weitergegeben werden können

CMU-TEK TCP/IP	(Campuslizenz)
PCSA	(15 Lizenzen für Institute)
T_EX	
SEDT	

TEKTRONIX-RISC-Workstation für ein Jahr am EDV-Zentrum

Walter Selos

Für die Unterstützung von Lehre und Forschung wurde von der Firma Tektronix eine RISC-Workstation XD88/20 leihweise für ein Jahr dem EDV-Zentrum zur Verfügung gestellt.

Diese basiert auf einem Motorola 88000 RISC-Prozessor, welcher bei einer Taktfrequenz von 20MHz eine Rechenleistung von 17 VAX-MIPS liefert.

Ein Grafik-Frontend ist mit einem 68020 ausgestattet und kann bis zu 90000 2D-Vektoren pro Sekunde verarbeiten. Das dazugehörige Display hat eine Auflösung von 1280*1024 Pixeln und 8 Bitplanes, was eine gleichzeitige Darstellung von 256 Farben aus einer Auswahl von 16.7 Millionen ermöglicht.

Als Schnittstellen stehen Futurebus, VME-bus und SCSI zur Verfügung. Ebenso sind ein Ethernet (802.3)-Anschluss sowie mehrere serielle Schnittstellen vorhanden.

Als Betriebssystem läuft UTekV, welches dem AT&T-UNIX SysV 3.2 entspricht. Es laufen darauf X-Windows, und die Systemschnittstelle entspricht dem POSIX-Standard.

Als Netzwerksoftware stehen TCP/IP (ftp,telnet,smtp-mailer), NFS, sowie uucp zur Verfügung. Alle Netzwerkdienste werden von uns verwendet und lassen eine problemlose Integration ins TU-interne Netz, wie auch Wide-Area-Networking über uucp (z.B. EUNET/Usenet) zu.

Zur Programmentwicklung stehen ein C-Compiler, Fortran77, sowie die in Unix üblichen tools zur Verfügung.

Die Graphic-Workstation steht in der Gußhausstraße 25, im 3. Stock allen Interessenten für ein erstes Kennenlernen, für Software- bzw. Performance-Tests zur Verfügung. Herr Selos, der zuständige Betreuer des EDV-Zentrums, ist gerne bereit, Hilfestellung zu leisten. Die Werbeeinschaltung der Firma Tektronix enthält eine Beschreibung des Einsteigermodells XD88/10, dessen Universitätspreis ein weiteres Argument darstellt, sich diese RISC-Maschine genauer anzusehen. Um weiteres Informationsmaterial anzufordern, bedienen Sie sich bitte der dem "Feedback" beiliegenden Karte "Tek-Info für Wissenschaft und Forschung".