Kurzüberblick über Entwicklung und Eigenschaften von PEARL im Vergleich mit anderen Programmiersprachen

P. Elzer

1. Einleitung

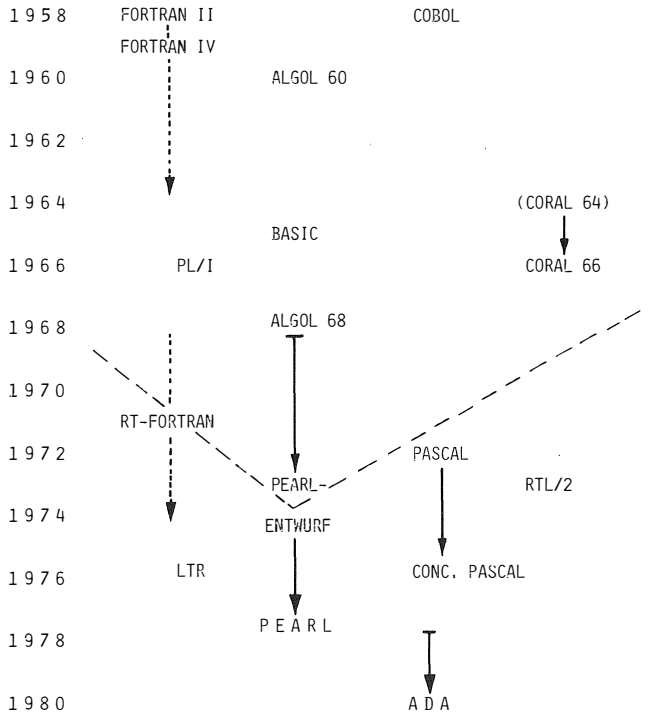
Auf der Jahrestagung des PEARL-Vereins im Dezember 1980 in Düsseldorf wurden auch zwei Vorträge gehalten über die Eigenschaften von PEARL im Vergleich mit den Programmiersprachen: Concurrent PASCAL, Ada, Realtime FORTRAN, CORAL 66, BASIC, RTL/2 und LTR. Diese Vorträge wurden im vollen Wortlaut in der PEARL-Rundschau Nr. 4, 1980 veröffentlicht [1,2], zusammen mit einer Darstellung der Entwicklungsgeschichte von PEARL durch den Verfasser [3]. Während seines Vortrages verwendete der Verfasser jedoch eine etwas andere Darstellungsweise und vor allem zwei Übersichtsbilder, nach denen er in der Zwischenzeit eine große Anzahl von Anfragen erhielt. Er möchte deshalb die Gelegenheit dieses Heftes nutzen, seine improvisierten Ausführungen von damals zu rekonstruieren und zu Papier zu bringen.

2. Die Entwicklung von PEARL

Diese wurde in [3] schon recht ausführlich dargestellt, wobei vor allem auch die Einflüsse anderer Programmiersprachen und parallel ablaufende Sprachentwicklungen berücksichtigt wurden. In Fig. 1 wird nur noch einmal versucht, einen gerafften Überblick in grafischer Form zu geben, da ja bekanntlich "ein Bild mehr sagt als tausend Worte". Ganz verkürzt könnte man sagen, daß alle Programmiersprachen oberhalb der gestrichelt dargestellten "Entwicklungsbugwelle" irgendeinen Einfluß auf die Entwicklung und die Eigenschaften von PEARL hatten, die anderen kaum.

3. Die Problemüberdeckung von PEARL

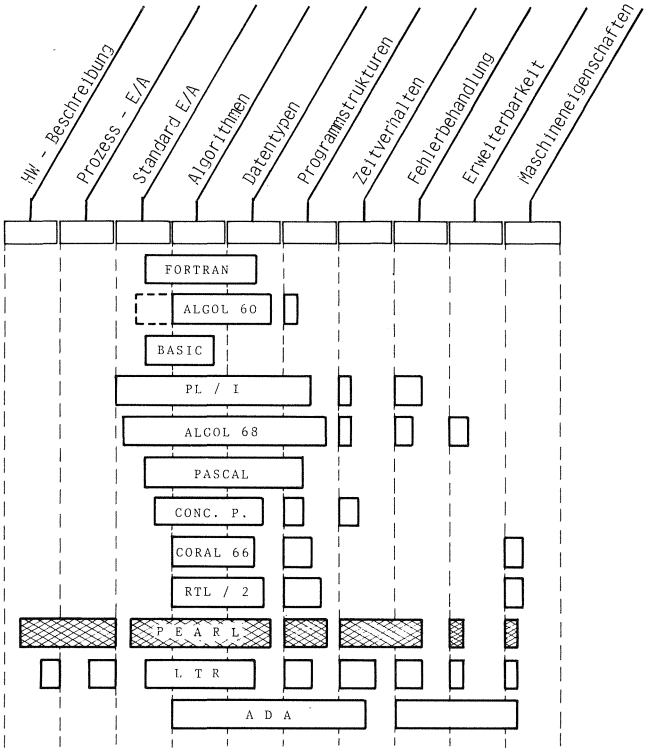
Dieser Ausdruck ist erläuterungsbedürftig. Normalerweise spricht man ja von "Elementen" oder "Eigenschaften" einer Programmiersprache. Diese Begriffe erscheinen aber für einen extrem kurzen Vergleich nicht recht brauchbar. Was den Anwender ja letztendlich interessiert, ist: "Welche Probleme kann ich mit der in Frage stehenden Programmiersprache denn nun eigentlich lösen?"



Figur 1: Die Entwicklung von PEARL im Umfeld anderer Programmiersprachen

Auf diese Frage geben aber Überlegungen bezüglich "Sprachelementen" und "-eigenschaften" immer nur indirekte Antworten. Es ist ja doch so, daß ein bestimmtes Problem jeweils durch verschiedene Sprachelemente gelöst (oder beschrieben) werden kann oder daß die "Eigenschaften" einer Sprache über ihre Problemlösungskapazität nicht viel aussagen, sondern mehr für eine Klassifizierung gemäß anderer wissenschaftlicher Kriterien geeignet sind.

Es wurde deshalb einmal versucht, die Anforderungen der Anwendungsklasse "Realzeitprogrammierung" in Teilgebiete aufzugliedern. Diese gruppieren sich um den ursprünglichen und eigentlich immer noch zentralen Zweck der Formulierung von Algorithmen. Die Beschreibung der Datentypen und die Behandlung von Standard-Ein-/Ausgabe (Text) kamen relativ frühzeitig dazu. Der nächste große Schritt war dann der Einbau von Sprachelementen zur Struk-



Figur 2: Die Problemüberdeckung von PEARL im Vergleich mit anderen Programmiersprachen

turierung von Programmen, zur Beeinflussung ihres Zeitverhaltens und zur Behandlung eventuell auftretender Fehler. Bei Sprachen für Realzeitanwendungen, wie PEARL oder LTR, war es dann nötig, die Beeinflußbarkeit des Zeitverhaltens stark zu verbessern und weitere Teilgebiete, wie Prozeß-Ein-/Ausgabe, Beschreibung der prozeßorientierten Hardware und Maschineneigenschaften zu überdecken.

Auch eine gewisse Erweiterbarkeit der Sprache erschien in Anbetracht der großen Problemvielfalt auf dem Realzeitsektor angebracht. Dafür konnte auf anderen Teilgebieten auf letzten Komfort verzichtet werden.

Dies soll die Darstellung in Fig. 2 veranschaulichen. Die Spalten stellen die einzelnen Teilgebiete dar. Ihre Breite wurde gleichmäßig gewählt, um keine Wertung zu implizieren. Der Grad der Überdeckung der Teilgebiete durch die verschiedenen Sprachen wird jeweils durch die Länge der einzelnen Balken symbolisiert. So liefert PEARL z.B. eine vollständige Beeinflußbarkeit des Zeitverhaltens von Programmen, aber nur die nötigsten Elemente zur Erweiterbarkeit. Bei Ada z.B. ist dies gerade umgekehrt, usf.

4. Schlußbemerkung

Der Verfasser hofft, mit dieser Kurzdarstellung einem oft geäußerten Bedürfnis entsprochen und den Lesern der PEARL-Rundschau einige neue Argumentationshilfen geliefert zu haben.

[13] Frevert,L.: "Vergleich von PEARL mit Concurrent PASCAL und Ada" PEARL-Rundschau Bd.1,4 S.61 1980

[2] Rzehak,H.: " Vergleich mit Programmiersprachen,die vor bzw. parallel zu PEARL entwickelt wurden" PEARL-Rundschau Bd.1, 4 S.65 1980

[3] Elzer, P.: " Rückblick auf die Entwicklung von PEARL" PEARL-Rundschau Bd. 1,4 S.58 1980