Auswirkungen des Geheimnisprinzips und der Modularisierung nach Parnas auf den Entwurf und die Entwicklung von Software und Programmiersprachen bis in die Heutige Zeit

Einleitung

Ein global agierendes Softwarehaus hat zwei Entwicklungsstandorte. Einen in der USA, einen in Indien. In den USA arbeitet Entwickler A, in Indien Entwickler B.
A hat nun eine komplexe Java- Klasse geschrieben, dabei aber alle Attribute und Methoden als „public“ deklariert.
B benutzt eine Vielzahl von Methoden der Klasse und greift direkt auf Attribute zu. Unter anderem benutzt er auch Methoden, die von A lediglich als Hilfsfunktion betrachtet werden.
Es ergeben sich nun eine Vielzahl von Szenarien, in denen sich das Vorgehen von B, sowie Entwurfsfehler sehr negativ auswirken. Doch davon im Verlauf dieser Arbeit mehr.

Das zentrale Thema dieser Arbeit ist die Modularisierung von Software, bzw. deren Umsetzung mittels des Geheimnisprinzips. Hierzu sind die Ursprünge und grundsätzlichen Überlegungen, und vor allem deren Umsetzung zu beleuchten.

Modularisierung gilt heute als unabdingbare Grundvoraussetzung für erfolgreiche Entwicklung und Wartung komplexer Softwaresysteme. Sie ist ein Grundprinzip objektorientierter Softwareentwicklung und Verteilter Systeme.
Ebenso erfordert die Entwicklung von Software an verteilten Standorten eine Beschränkung der von den Entwicklern benötigten Informationen, also eine Anwendung des Geheimnisprinzips.

Das Geheimnisprinzip geht zurück auf David Lorge Parnas, geboren am 10. Februar 1941 in Plattsburgh New York. Er ist ein bekannter Softwaretechnikpionier.
Parnas erhielt seinen PhD. von der Carnegie Mellon University. Dort arbeitete er auch mehrere Jahre. Außerdem lehrte er an der University of North Carolina, der Technischen Hochschule Darmstadt und der University of Victoria. Zurzeit arbeitet er an der University of Limerick in Irland. In den 80igern Arbeitete eine kurze Zeit am SDI- Programm mit, kündigte dann aber bald seine Mitarbeit auf und war fortan der schärfste Kritiker dieser Art von Raketenabwehr. Er postulierte, dass ein derart hochgradig verteiltes System niemals fehlerfrei arbeiten könne.[][]
Seine Ideen bilden die hauptsächliche Grundlage der heutigen objektorientierten Programmiersprachen.

Im Folgenden Kapitel werden wir uns das Geheimnisprinzip in seiner Entstehung, sowie die Gründe, die zu seiner Entstehung führten näher betrachten.