Entwicklung einer modularen Software für dezentrale, verteilte Berechnungen nach der FE-Methode

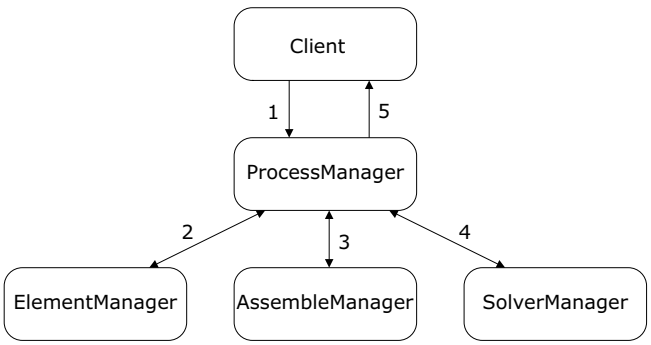
Diplomarbeit von André Lähr
entstanden am Lehrstuhl für Statik der TU München

Verteiltes Rechnen

Das Ziel ist es einen Prozess in mehrere Arbeitsschritte aufzuteilen und diese auf verschiedene Rechner zu verteilen. Je höher der Verteilungsgrad eines solchen Prozesses ist, desto mehr wirken sich die positiven Eigenschaften des Verteilten Rechnens auf die Funktion aus. Der Verteilungsgrad hängt direkt davon ab, den ursprünglichen Vorgang in möglichst viele kleine, voneinander unabhängige Arbeitsschritte aufzuteilen.
Durch entsprechende Organisation können die entstandenen Teilprozesse sequentiell und falls möglich auch parallel abgearbeitet werden. Vorteilhaft dabei ist, dass das Hinzuziehen vieler (auch kleiner) Rechenkapazitäten die Leistung des Systems erhöhen können. Man ist also nicht unbedingt darauf angewiesen lokal leistungsstarke Ressourcen bereitzustellen, um die gewünschte Aufgabe zu erfüllen.

Das Programm (DFEC)

DFEC steht für Distributed Finite Element Computing. Es folgt eine grobe Beschreibung eines Berechnungsablaufs. Die Abbildung stellt einen Überblick über die Struktur der Software dar:



Der Client (Anwender) übergibt (Punkt 1) die notwendigen Parameter (Geometrie, Material, usw.) für eine FE-Berechnung an einen ProcessManager. Der ProcessManager verarbeitet die Parameter und leitet die sich daraus ergebenden Daten an einen ElementManager weiter. Über diese Beziehung werden Daten bezüglich der Elemente (im FE-Sinne) ausgetauscht. D.h. der ProcessManager erhält die lokale Steifigkeitsmatrizen der Elemente als Ergebnis zurück (Punkt 2). Diese werden wiederum vom ProcessManager weiterverarbeitet und an einen AssembleManager übergeben. Er ist für den Zusammenbau der globalen Steifigkeitsmatrix verantwortlich, die er wieder an den ProcessManager sendet (Punkt 3). Ein SolverManager erhält nun als letzten FE-Schritt die notwendigen Informationen zum Lösen des Gleichungssystems vom ProcessManager und liefert ihm das Ergebnis (die Verschiebungen) zurück (Punkt 4). Der ProcessManager schickt dem wartenden Client das Ergebnis (Punkt 5).
Jeder dieser einzelnen Komponenten ist selbständig und kann auf einem separaten Computer ausgeführt werden.

Erreichte Funktionalität

Wie bei herkömmlichen FE-Programmen besteht die Möglichkeit das Programm zu erweitern, ohne die Struktur der Software verändern zu müssen. Als Datenaustauschformat wurde XML (eXtensible Markup Language), ein standardisiertes und verbreitetes Format, realisiert. Auch hat sich der Einsatz von Java hauptsächlich für die Netzwerkkomunikation als günstig erwiesen.
Zusätzlich wurde eine wertvolle Eigenschaft, die vor allem urheberrechtliche Aspekte berücksichtigt, geschaffen. Dadurch können Algorithmen (bzw. Code) besonders geschützt werden, da dem Anwender nur deren Funktionalität für eine Berechnung zur Verfügung gestellt wird.

Probleme

Durch das Verschicken der Eingabe- und Berchnungsdaten zwischen den einzelnen Komponenten entsteht ein sehr hoher Bedarf an Netzwerkbandbreite. Dies führt zu sehr großen Verzögerungen bei einer komplexen Berechnung.
Java hat einen Performancenachteil gegenüber nativen Programmiersprachen. Es besteht jedoch die Möglichkeit an für rechenintesive Vorgänge andere, schnellere Sprachen einzusetzen.
Ein weiteres Problem bei solchen Netzwerkanwendungen stellt die Sicherheit dar (Hackerangriffe, usw.). Um die Funktionalität und evtl. Datenschutzaspekte zu gewährleisten darf dieser Punkt unter keinen Umständen vernachlässigt werden.