Team: TEAM 01, Falco Winkler (FW), Daniel Schruhl (DS)

Aufgabenteilung:

- Starter (DS)

Quellenangaben:

- Aufgabe 2, 25.04.2017, C.Klauck:
  http://users.informatik.haw-hamburg.de/ klauck/VerteilteSysteme/aufg2.html

Bearbeitungszeitraum:

- 13.04.2017 3h (FW)
- 24.04.2017 3h(DS)
- 25.04.2017(DS)

Aktueller Stand:

- Starter-Modul fertig und getestet
- ggT-Modul angefangen
- Koordinator-Modul angefangen

Änderung des Entwurfs:

- Keine Änderungen

1 Einführung und Ziele

Mit dem Satz von Euklid ist es möglich, den größten gemeinsamen Teiler (ggT) zweier positiver ganzer Zahlen ($x, y \in \mathbb{Z}_+^*$) zu bestimmen (Gleichung 1).
Dabei wird der größte gemeinsame Teiler von x und y auf den größten gemeinsamen Teiler von y und $\bmod(x, y)$ zurückgeführt.

$$\forall x, y \in \mathbb{Z}_+^* : ggT(x, y) = ggT(y, \bmod(x, y)) \tag{1}$$

Das erlaubt eine rekursive Berechnung des größten gemeinsamen Teilers.
Das Produkt soll diesen Algorithmus verteilt ausführen, verwalten und koordinieren, um den größten gemeinsamen Teiler zu berechnen.

1.1 Randbedingungen

Um den ggT verteilt mit dem Algorithmus berechnen zu können, muss der Algorithmus angepasst werden (Gleichung 2). Das ermöglicht ein Terminieren in jedem ggT-Prozess mit dem ggT und nicht mit 0, da bei $ggT(x, x)$ terminiert wird.

$$\forall x, y \in \mathbb{Z}_+^* : ggT(x, y) = ggT(y, \bmod^*(x, y))$$
$$\bmod^*(x, y) := \bmod(x-1, y) + 1 \tag{2}$$

1.2 Kontextbegrenzung

Das System soll in Erlang umgesetzt werden. Es muss auf Computern mit Linux Betriebssystem lauffähig sein.

2 Gesamtsystem

2.1 Bausteinsicht

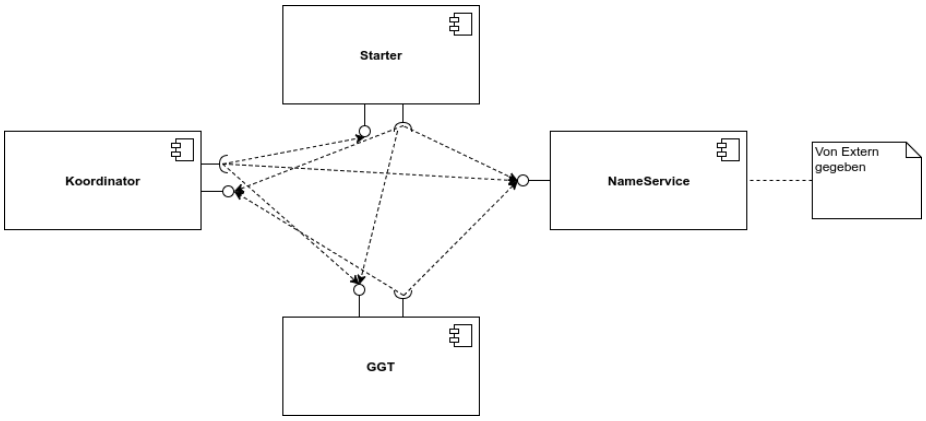

Abbildung 1: Komponentendiagramm der ggT-App

2.2 Laufzeitsicht

3 Subsysteme und Komponenten

3.1 Starter-Modul

3.1.1 Aufgabe und Verantwortung

Der Starter steht zwischen Koordinator und ggT-Prozess. Er startet mehrere ggT-Prozesse mit den Initialisierungsdaten, die vom Koordinator zur Verfügung gestellt werden (asynchron) und seinen konfigurierbaren Parametern.

3.1.2 Schnittstelle

```
/* Nachricht zum Starten von ggT-Prozessen */
{steeringval, ArbeitsZeit, TermZeit, Quota, GGTProzessnummer}


/* Startet einen Starter Prozess */
start(StarterID): Integer -> PID
```

{**steeringval**,ArbeitsZeit,TermZeit,Quota,GGTProzessnummer}:
Diese Schnittstelle wird für den Koordinator bereitgestelt und ermöglicht die Übergabe benötigter Konfigurationsparameter, nach der Anfrage des Starters über getsteeringval. Nach dem Ansprechen dieser Schittstelle sollte der Startvorgang der GGT Prozesse eingeleitet werden. Die ArbeitsZeit ist die simulierte Verzögerungszeit zur Berechnung in Sekunden, die TermZeit ist die Wartezeit in Sekunden, bis eine Wahl für eine Terminierung initiiert wird, Quota ist die konkrete Anzahl an benötigten Zustimmungen zu einer Terminierungsabstimmung und GGTProzessnummer ist die Anzahl der zu startenden ggT-Prozesse.

start(StarterID):
Startet den Starter mit der gegebenen eindeutigen Nummer.

3.1.3 Entwurfsentscheidungen

Der Starter ruft in seiner Initalisierungsphase an der Schnittstelle vom Koordinator seine benötigten Parameter zum Starten von ggT-Prozessen asynchron ab. Zusätzlich werden noch die konfigurierbaren Parameter geladen. Mit diesen Daten werden dann die ggT-Prozesse gestartet. Die Anzahl der zu startenden ggT-Prozesse wird durch die GGTProzessnummer in der Schnittstelle definiert. Der Koordinator hält keine Referenz zu Starter - Prozessen, da nur die ggT Prozesse für Ihn relevant sind.

3.1.4 Konfigurationsparameter

- Praktikumsgruppe
- Teamnummer
- Nameservicenode definiert den Node des Nameservices

- Nameservicename definiert den Namen des registrierten Nameservices auf dem Nameservicenode
- Koordinatorname definiert den Namen des Koordinators

3.2 ggT-Modul

3.2.1 Aufgabe und Verantwortung

TODO

3.2.2 Schnittstelle

```
/* Einkommende Nachricht zum Setzen der Namen der Nachbarn */
{setneighbors, LeftN, RightN}

/* Einkommende Nachricht zum Setzen der von diesem Prozess */
/* zu berabeitenden Zahl fuer eine neue Berechnung */
{setpm, MiNeu}

/* Einkommende Nachricht zum Senden des rekursiven Aufrufes der ggT Berechnung */
{sendy, Y}

/* Einkommende Wahlnachricht fuer die Terminierung der aktuellen Berechnung */
{From, {vote, Initiator}}

/* Einkommende Erhaltenes Abstimmungsergebnis */
{voteYes, Name}

/* Einkommende Nachricht zum Senden des aktuellen Mis an From */
{From, tellmi}

/* Einkommende Nachricht zum Senden eines pongGGT an From */
{From, pingGGT}

/* Einkommende Nachricht zum Beenden des ggT-Prozesses */
kill

/* Funktion, die einen ggT-Prozess startet */
start(WorkingTime, TerminationTime, Quota, GGTName, Coordinator, NameService):
Integer X Integer X Integer X Atom X Tupel X PID -> PID
```

{**setneighbors**,LeftN,RightN}: Setzt die Nachbarn. LeftN und RightN sind dabei Namen, die im NameService (und lokal im Node) registriert sind.

{From,{**vote**,Initiator}}: Wahlnachricht für die Terminierung der aktuellen Berechnung. Der Initiator ist der Initiator dieser Wahl (Name des ggT-Prozesses, keine PID!) und From (ist PID) ist sein Absender.

{**voteYes**,Name}: Erhaltenes Abstimmungsergebnis, wobei Name der Name des Absenders ist (keine PID!).

{From,**tellmi**}: Sendet das aktuelle Mi an From (ist PID): From ! {mi,Mi}. Wird vom Koordinator z.B. genutzt, um bei einem Berechnungsstillstand die Mi-Situation im Ring anzuzeigen.

{From,**pingGGT**}: Sendet ein pongGGT an From (ist PID): From ! {pongGGT,GGTname}. Wird vom Koordinator z.B. genutzt, um auf manuelle Anforderung hin die Lebendigkeit des Rings zu prüfen.

start(WorkingTime, TerminationTime, Quota, GGTName, Coordinator, NameService**)**: Startet einen ggT-Prozess mit den gegebenen Parametern. Der GGTName setzt sich zusammen aus <PraktikumsgruppenID><TeamID><Nummer des ggT-Prozess><Nummer des Starters>. Die WorkingTime beschreibt einen simulierten Arbeitsaufwand für die Berechnung und die TerminationTime beschreibt die Zeit, nach der ein ggT-Prozess eine Terminierungsabstimmung durchführt. Die Quota ist die konkrete Anzahl an notwendigen Zustimmungen zu einer Terminierungsabstimmung. Coordinator und NameService sind Referenzen für die jeweiligen Dienste.

3.2.3 Entwurfsentscheidungen

Das Modul hält sich seinen State mit einer Config-Map. In dieser Map sind alle benötigten Variablen für den Algorithmus und für den Betrieb des ggT-Moduls gespeichert.

3.3 Koordinator-Modul

3.3.1 Aufgabe und Verantwortung

Der Koordinator verwaltet alle ggT Prozesse. Er kommuniziert mit Starter-Prozessen um diesen die benötigten Werte zum Starten der ggT-Prozesse zu übergeben. Alle ggT-Prozesse müssen sich außerdem bei ihm anmelden, und er übernimmt die Anordnung dieser in einem Ring, darüber hinaus die Terminierung des gesamten Systems auf Befehl eines ggt-Prozesses.
Im Koordinator - Modul werden die drei Zustände durch drei receive - Schleifen realisiert. Alle Referenzen auf GGT Prozesse werden in einer Erlang - Liste persistiert. Meldet sich ein GGT - Prozess beim Koordinator, wird er am Ende dieser Liste angefügt.
Im Zustandsübergang zu "bereit"wird die Liste der GGT-Prozessnummern gemischt, iteriert, und die Knoten bekommen ihre entsprechenden Nachbarn zugewiesen.

3.3.2 Schnittstelle

3.3.3 Entwurfsentscheidungen

3.3.4 Konfigurationsparameter