Big Bang Challenge

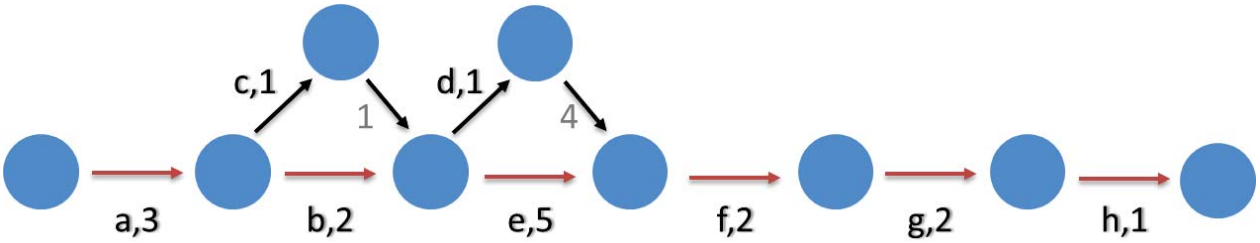
Aufgabe 1

Teamname: unknown.users

Teammembers:

Aufgabe a):

1) „Plangetriebene“ bzw. „klassische“ Softwareentwicklung bedeutet, dass die einzelnen Entwicklungsphasen am Anfang des Projekts geplant werden und dann nacheinander umgesetzt werden.
   Bei der „agilen“ Softwareentwicklung wird versucht, möglichst flexibel zu arbeiten. Dies geschieht durch ständige Rücksprache im Team und mit dem Kunden. Am Anfang wird noch kein konkreter Plan festgelegt.
   Die „hybride“ Softwareentwicklung versucht die anderen beiden Methoden so zu kombinieren, dass ein möglichst gutes Verhältnis zwischen Flexibilität, Kundenzufriedenheit und Schnelligkeit geschaffen wird. Dies geschieht durch eine Gesamtplanung am Anfang, bei der auch später noch Änderungen möglich sind.
2) Vorteil der „agilen“ Softwareentwicklung ist die hohe Flexibilität. Dadurch kann schnell auf Probleme und Änderungswünsche eingegangen werden. Gleichzeitig kann dieses Fehlen eines anfänglichen Gesamtproblems auch ein Nachteil sein, denn so können Zeit und Kosten am Anfang nicht gut eingeschätzt werden. Vorteile der „plangetriebenen“ Softwareentwicklung ist die klare und einfache und somit auch nachvollziehbare Struktur. Durch das Festlegen der Pläne am Anfang kann, anders als bei der „agilen“ Softwareentwicklung, nicht mehr so gut auf Änderungswünsche eingegangen werden. Fehler werden erst (zu)spät entdeckt und zusätzliche Kosten entstehen.
   Bei der „hybriden“ Softwareentwicklung werden sich die eben gebannten Vorteile so angeeignet, dass am Anfang ein klarer Plan entsteht, später aber auch noch Dinge verändert werden können, um möglichst schnell und ohne zusätzliche Kosten zu entwickeln. Es entstehen möglichst wenige Nachteile.
3) Kritischer Pfad:



4) Die kürzeste Dauer des Projekts führt über den kritischen Pfad:
   $a + b + e + f + g + h = 15\ Wochen$

   Die Puffer entsprechen den Scheinpfaden im Netzplan, daher gibt es für c einen Puffer von einer Woche und für d einen Puffer von 4 Wochen.

5) Wenn jeder Pfad von einer Person durchgeführt werden kann und nicht für eine Aufgabe mehrere Personen benötigt werden, kann das Projekt mit zwei Mitarbeitern frühestmöglich fertiggestellt werden, da nur b und c sowie d und e parallel laufen.

Aufgabe b):

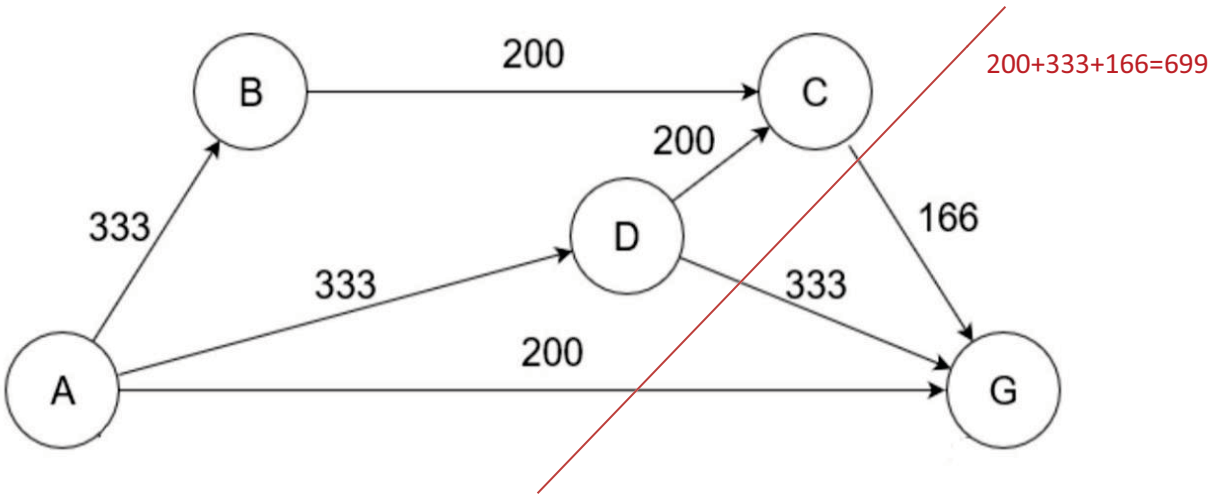
1) Software-Entwicklung erfordert Teamarbeit, da durch die Aufteilung von Aufgaben Zeit gespart werden kann. Dadurch, dass nicht eine einzige Person den Überblick über das ganze Projekt hat, führt Teamarbeit auch zu einer besseren Übersichtlichkeit. Außerdem kommt man so auf verschiedene Lösungsansätze und kann Probleme besser lösen.
2) Bei kleineren Projekten kann es besser sein, wenn ein einzelner Entwickler arbeitet, da man hier alle Informationen sofort erhält und keine „verloren" gehen. Außerdem spart man Zeit, da man keine Rücksprachen mit Kollegen halten muss. Stattdessen kann man selbstständig und somit auch schneller und flexibler arbeiten.
3) Zu festen Informationsspeichern gehören Zeitungen, Bücher und CDs. Flüssige Informationsspeicher sind Menschen, Notizen und bestimmte Nachrichten, z.B. auf Snapchat.
   E-Mails sind zwar langfristig und wiederholt aufrufbar und können für Dritte verständlich sein und somit zu festen Informationsspeichern gehören. Einige E-Mails, beispielsweise an Freunde, sind jedoch nicht für alle verständlich und gehören somit zu flüssigen Informationsspeichern.
4) Über diese Knoten werden besonders viele Informationen übertragen, weshalb sie sehr wichtig für das Projekt sind. Sollte einer dieser Knoten ausfallen, könnte das Projekt viel schwieriger umzusetzen sein bzw. länger dauern, da dann die Informationen über „Umwege" fließen müssen. Sollte dies nicht möglich sein, kann es auch dazu kommen, dass das Projekt nicht mehr umsetzbar ist.
5) Ein Zentralitätsmaß ist Betweenness. Ein Knoten hat eine besonders hohe Betweenness, wenn er Bestandteil besonders vieler kürzester Wege ist. Der kürzeste Weg zwischen zwei Knoten ist der Weg, über die man vom Startknoten zum Zielknoten kommt und dabei so wenig andere Knoten wie möglich passiert.
   Bei der Closeness-Zentralität wird die Nähe zu anderen Personen berücksichtigt, also wie gut ein Knoten zu anderen Knoten auf direktem Weg verbunden ist.
   Der Degree gibt an, über wie viele direkte Pfade dieser Knoten verbunden ist. Dabei wird zwischen Indegree, der Anzahl der eingehenden Pfade, und Outdegree, der Anzahl der eingehenden Pfade unterschieden.
6) Die zentrale Position der Berater lässt sich mit allen genannten Zentralitätsmaßen belegen. Die Berater haben den höchsten Degree, mit einem Indegree von 8 und einem Outdegree von 6. Dadurch haben sie auch eine große Nähe zu allen anderen Personen und einen hohen Closeness-Wert. Außerdem führt der hohe Degree auch zu einer besonderes großen

Betweenness, da die Berater durch die vielen Verknüpfungen auch auf besonders vielen Pfaden liegen.

7) Im Vergleich zu den Beratern haben die anderen Knoten nur eine geringe Zentralität. So haben der Teamleiter Berater sowie der Geschäftsführer einen Degree von 3, die Berater einen von 14.
   Außerdem gibt es sehr viele „doppelte“ bzw. überflüssige Pfade, die das Diagramm viel umständlicher machen. So können Berater und Entwickler z.B. direkt und über den Teamleiter Informatiker kommunizieren.
8) Die Teamleiter könnten einige Aufgaben der Berater übernehmen, sodass es nicht so schlimm ist, falls diese ausfallen sollten. Außerdem könnte man eine direkte Verbindung zwischen den beiden Teamleitern einrichten, damit diese schneller und flexibler ohne den Umweg über die Berater kommunizieren.

Aufgabe c):

1) Der maximale Fluss ist die Gesamtkapazität zwischen dem Start- und Zielpunkt, also wie viele Informationen maximal zwischen den beiden Punkten fließen können.
2) Als Erstes muss der Start- und Zielknotenpunkt feststehen. Anschließend geht man vom maximalen Übertragungswert vom Zielpunkt aus, die auf die nächsten Punkte übergehen. (siehe Abbildung: A zu B 333; A zu D 333; A zu G 200). Anschließend sieht man sich die kleinsten Übertragungswerte auf den einzelnen Pfaden von Anfang bis Ende an. Auf der Route {A;B;C;G} ist der kleinste Wert 166.
   Route {A;D;G} ist der kleinste Wert 333.
   Route {A;G} ist der kleinste Wert 200.
   Daraus kann man schlussfolgern, dass der maximale Wert, der ankommen kann, bei 699 liegt. Anschließend sieht man sich den Anfangspunkt an und überprüft, ob dieser Wert 699 erreicht werden kann. Das ist hier in diesem Falle richtig. Mit 333+333+200= 866. Somit decken sich die Ergebnisse.
3) Der maximale Fluss vom A zu G beträgt 699.
4) Der minimale Schnitt im Diagramm ist der Schnitt mit den geringsten Kantengewichten, also dort, wo die Pfeilwerte, die geschnitten werden, die geringste Summe haben.
5)



6) Das max-flow-min-cut-Theorem besagt, dass der maximale Fluss im Diagramm genau den Wert des minimalen Schnitts hat. Dadurch kann man Schwachstellen in Netzwerken einfacher entdecken und z.B. Logistiksysteme verbessern.