Assist.Network

Rendszer logikai működése

Az Assist.Network a segítői hálózat fizikai logisztikai folyamatokat támogatja.

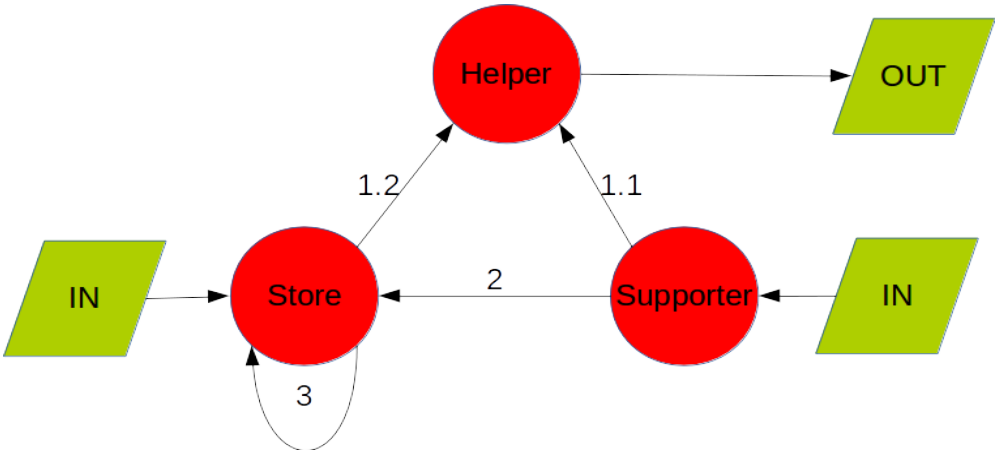
A rendszer bevezetését két fázisban szeretnénk megvalósítani, melyben a támogatási cél:

1. Kommunikáció

2. Optimalizáció

A rendszer egy teljesen elosztott Tároló (Store), amely képes azt az információt szolgáltatni, hogy az adott pontján a hálózatnak miből mi és mennyi van, azaz miből van hiány és hol van „felesleg”. Az Assist.Network logisztikai célja, hogy a Node-okon, ahol hiány van, a hiány csökkenjen.

A Helper Node-ok a hálózatból árut „visznek ki” a Supporter és a Store node-ok árut „hoznak be”. A Store annyiban különbözik a Supporter node-tól, hogy majd a későbbiekben (V2) készletet is nyilván tart, valamint „hozzáadott értékű” szolgáltatást is végez (csomagolás).



Fizikai árumozgások az Assist.Networkben

1. Helper Igényel
2. Store Offer-t fogad a Supportertől.
3. Ez a legizgalmasabb, Store tud másik Storetól is fogadni. (Később megy majd a Push is, ekkor lehet majd optimalizálni)

Az Assist.Network a valóságban egy olyan gráf, amiben az egymáshoz közel lévő node-ok vannak összekötve. Az egymáshoz közelséget az határozza meg, ahol megéri az adott árut fuvarozni (pl egy polót BP-ről Szegedre). Mivel a hálózatban jelenleg a túltelítés a jellemző, ezért elsődlegesen (amíg a rendszer indul, és nem tudunk optimalizálni) PULL architektúrát alkalmazunk, azaz minden Node a hálózatban akkor kérhet árut, amikor szüksége van (Helper), vagy tárolni tudja (Store), Supporter (amíg el nem szállítják tőle). Ezért minden elem, valójában egy

Gyűjtő/Collector sémából származtatható.

A termék/szolgáltatás „fizikai” mozgása az Assist.Networkben.

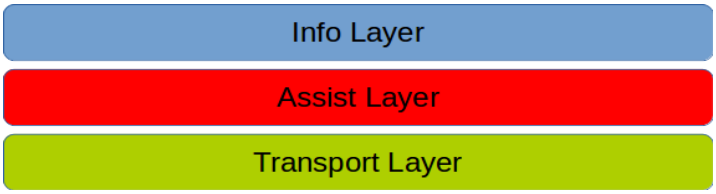
OUT ← Helper ←[Store || Supporter] ← IN

Minden user a hálózatban egy node, aminek két „oldala” van: demand, és a supply. Azaz a szükséglet illetve a kínálat. De valójában ez a kettő oldal az egy vektor, aminek a negatív értéke a demand, a pozitív a supply.
Tehát az Assist.Network az {n.listings[]} vektortér. A vektor bázisa a termékek és szolgáltatások halmaza a hálózatban. Ez a leírásmód egy adott Node-on (User) belül is képes akár egyszerre Offer-t és Demand-ot is mutatni különböző termékekre! (pl cipő #női #35 -3db , miközben cipő #női #37 5db).

Ez a modell képes teljes elosztottságra, ugyanis lehet az igények kielégítését az összes Node-re végezni (ez nem racionális, hisz a szállítási költséget és időtartamot nem veszi figyelembe) lehet a távolság és költség figyelembe vételével, illetve lehet szomszédokra, vagy hálózati módon akár ismerősi kapcsolatokra is alapozni (ez a valóságos működés az adományozás során).

A prototípusban az Igény „kielégítésénél” csak adott távolságon belüli Node-ok supply-jait vesszük figyelembe.

A node-os kezelés lehetőséget ad könnyen skálázható rendszer kiépítésére, de a teljesen elosztott, központi elem nélküli (teljesen robusztus) működésre is. Egy havária esetén (adott időben, adott helyszínen szükség van segítségre), a rendelkezésre álló supply kiszámítása a racionális távolságú node-ok potenciáljának kiszámításával keletkezik (statisztikusan értelmezhető)



A rendszerben három réteget különböztetünk meg:

- Info layer
- Assist layer
- Transport réteg

A Layerek értelme, hogy az Assist Layerben „áramlik” a Demand és Offerek „árama”, míg a Transport Layerben a valódi fizikai termékek és szolgáltatások árama.

Igaz a megmaradás törvénye, azaz a

Az Info layerből lehetséges Demand-ot generálni az Assist layerbe (Reporter dobhat Demandot Helpernek, vagy csak a Helper készíthet Infóból Demandot?). Jelenleg nem foglalkozunk az Info layerrel (ahol a spotting, texting majd szabadon megy), mert erre van több párhuzamos alkalmazás fejlesztés, és a segítő szervezetek jellemzően a logisztikai koordináció hiányát említették a legfőbb problémának.

Assist Layer-t az előző részben kifejtettük a Demand-Supply kezelésben. Az igények kielégítése azonban fizikai mozgatási igényt generál (Transport Task – TT). Ezért az Assist Layer a Transport Taskot a Transport Layernek küldi úgy, hogy ha a Helper vállalja szállítást, akkor a Feladatot a Helperre szignálva, ha ő nem tudja végezni, akkor a Supporter, vagy Store vállalása alapján nekik szignálva. Azonban gyakori eset, hogy a tranzakcióban szereplőknek nincs megfelelő, vagy szabad Szállítói Kapacitásuk. Ekkor a Transport Task a Transport Layerben található „releváns" Szállítók számárai láthatóvá válik, és az első szállító, aki elfogadja a Helper és Supporter szállítási időkorlátját, és elvállalja a Feladatot, az kapja. A V1-ben ennek csak a kommunikációját kezeljük, a V2-ben Optimalizálásra is lesz mód.

API

A userek regisztrációkor kapnak egy NodeID-et. Ez az ID az ő logikai azonosítójuk.

Innen a Node és User-t azonos Entitásként kezeljük.

Az API REST megvalósítása a következő:

http://assist.network/ - base URL

login

sign

post – ez lesz majd a texting, spotting interfész (post)

info – ez lesz az általános infó rész (itt tölti le a híreket, ide jöhetnek az rss-ek, stb)

assist/:nodeID # lehet nodeID paraméter (Mr. Golda? - itt sec-et is érdemes, főleg ha a node egy szép hash)

demand post {package} , result demandID – Igény bejelentés

offer post {package} , result offerID – Felajánlás

listings get – a Node-on a Demand-ok és Offerek összessége.

collect post { package }, result collectID – Offer/Felajánlás begyűjtése , ebből keletkezik ha nincs szállítás a Felajánlótól, vagy a Gyűtőtől, akkor ez megy a Transport Layerbe.

transport/

task get {location}

bind post {vehicleID, orderID }

track get {vehicleID}

vehicles {nodeID}