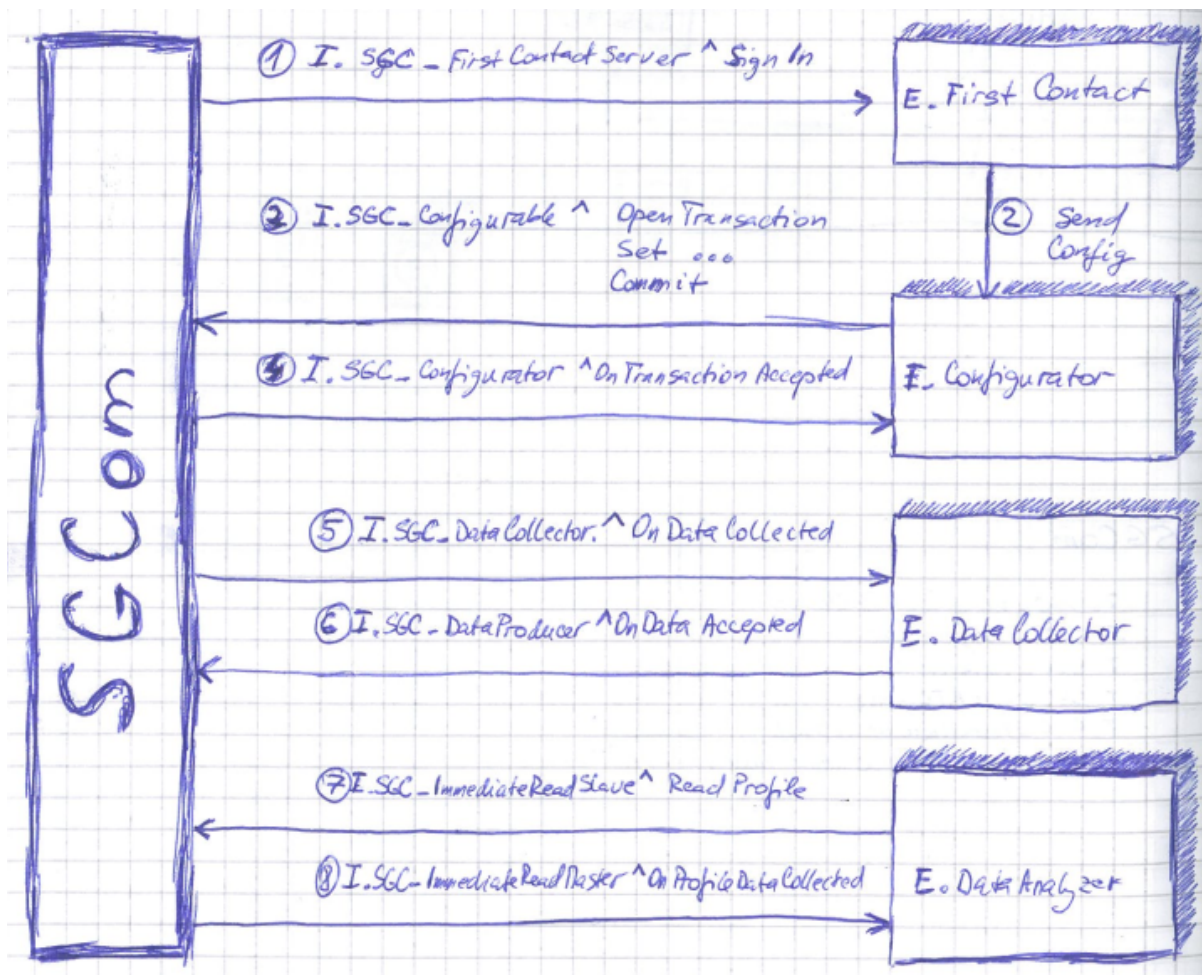


# Životný cyklus SGCom-u

SGCom je navrhnutý tak, aby bol odolný voči rôznym formám výpadkov. Pri svojom štarte sa musí pripojiť k centrále, aby získal konfiguráciu (bez ktorej nevykoná žiadnu akciu), ale potom môže pracovať aj v prípade, že stratí s centrárou na nejaký čas kontakt. Je robustný aj v prípade, že nedokáže vykonať zber dát, o čom informuje centrálu a poskytuje mechanizmy, ako neskôr dáta opätovne získa. Životný cyklus ilustruje nasledujúci obrázok a možno ho tiež popísať v nasledovných krokoch.



Obrázok 3 Ilustrácia životného cyklu

- Štart aplikácie, spracovanie parametrov príkazového riadku, ktoré okrem iného obsahujú adresu pre pripojenie sa ku D2000 kernelu, prihlasovacie údaje a adresu tzv. „bodu prvého kontaktu“.
- Vykonanie údržby modulu *DataStorage*:
  - Vymazanie záznamov starších ako 90 dní. Úloha sa opakuje každých 24 hodín.
  - Odstránenie označenia úspechu transakcie zo všetkých záznamov (podrobnosti v kroku 7).
- Pripojenie ku D2000 kernelu centrálne. SGCom sa pokúša získať D2Japi Session, ktorá bude v DODM reprezentovaná objektom s názvom v tvare SGCOM1.SGC. V prípade neúspechu sa pokus o pripojenie opakuje každých 30 sekúnd.
- Prihlásenie sa do „bodu prvého kontaktu“. Adresa objektu je definovaná parametrom z príkazového riadku –f., o je inštancia objektu typu D2000 Event, ktorá implementuje rozhranie *I.SGC\_FirstContactServer*. Prihlási sa volaním RPC *SignIn* s hodnotou parametra *\_name* nastavenou na meno procesu (z príkazového riadku). RPC je asynchrónne a zakladá aplikované riadenú transakciu, ktorá zabezpečí, že sa buď SGCom alebo centrála dozvie, keď jej partner havaruje.
- Centrála na nadviazanie prvého kontaktu reaguje tým, že odošle do SGCom-u novú konfiguráciu, resp. zmení existujúcu. D2000 Event, ktorý riadi zasielanie konfigurácie musí implementovať rozhranie *I.SGC\_Configurator*, SGCom pre prijatie implementuje rozhranie *I.SGC\_Configurable*.
  - Prenos konfigurácie je transakčný. Všetky zmeny sú buď v poriadku a prijaté ako celok, alebo sú všetky zamietnuté a konfigurácia je ponechaná v pôvodnom stave. Postupnosť krokov je nasledovná:
    - Otvorenie transakcie: *I.SGC\_Configurable^OpenTransaction*
    - Odoslanie zmien: *I.SGC\_Configurable^Set\** a *Delete\**.
  - Príkazy je nutné zadávať v takom poradí, aby nebola v žiadnom okamihu porušená referenčná integrita konfiguranej databázy.
    - Commit transakcie: *I.SGC\_Configurable^Commit*
    - Potvrdenie prijatia zmien *I.SGC\_Configurator^OnTransactionAccepted*, alebo zamietnutie zmien *I.SGC\_Configurator^OnTransactionDenied* so zoznamom chýb v parametri.
  - Alternatívne môže byť transakcia zrušená zo strany centrálne volaním *I.SGC\_Configurable^Rollback*, o SGCom potvrdí *I.SGC\_Configurator^OnTransactionRolledBack*. Transakcia je automaticky zrušená aj v prípade, že sa preruší aplikované riadená konverzácia.

6. SGCom podľa aktuálnej konfigurácie SGCom nastaví svoj harmonogram úloh – pre každú entitu *PeriodicEvent* vytvorí zvlášť záznam a naplánuje jej vykonanie.
7. Keď nastane asový okamih, na ktorý je naplánovaná periodická úloha, SGCom prezrie aktuálnu konfiguráciu a zane vykonáva všetky súvisiace úlohy. Do modulu *TaskExecutor* vloží do príslušných front úlohy na zber a odovzdávanie dát, ktoré *TaskExecutor* následne vykoná. Vykonanie úloh, ktoré si vzájomne nekonkurujú je paralelné. (Pozn.: vzájomne si konkurujú dve úlohy, ktoré sú vykonané na tom istom logickom zariadení a tiež úlohy, ktoré obsluhujú logické zariadenia pripojené tou istou komunikačnou linkou. Zber a odovzdávanie dát si vzájomne nekonkurujú.) Zber dát každej jednej veličiny, i už úspešný alebo neúspešný, končí tým, že je výsledok zaznamenaný do modulu *DataStorage*. Pri úspešnom zbere je to zoznam hodnôt s asovými znakmi, pri neúspešnom je to kód a popis chyby, tiež s asovou značkou. Odovzdanie hodnôt centrálne prebieha po jednotlivých logických zariadeniach a jednotlivých veličinách (*DataPoint*). Každé odovzdanie predstavuje samostatnú transakciu a až keď centrála potvrdí úspešné prevzatie hodnôt, sú dáta odstránené z *DataStorage*. Odovzdanie pre jedno logické zariadenie a veličinu pozostáva z nasledujúcich krokov:
  - a. Vygenerovanie unikátneho ID transakcie.
  - b. Všetky záznamy pre dané logické zariadenie a veličinu sú oznaené íslom transakcie. (V prípade, že predchádzajúca transakcia neskončila, sú oznaené len nové záznamy.)
  - c. Hodnoty a asové znaky oznaených záznamov sú odoslané volaním *I.SGC\_DataCollector.OnDataCollected*.
  - d. Po spracovaní dát je transakcia ukonená volaním *I.SGC\_DataProducer.OnDataAccepted*.
  - e. Záznamy oznaené príslušným íslom transakcie sú vymazané z *DataStorage*-u.
 (Pozn.: RPC komunikácia prebieha v rámci aplikácie riadenej transakcie, takže ak spracovanie odovzdaných dát z nejakého dôvodu zlyhá, SGCom sa túto skutočnosť dozvie a transakcia sa zruší. Dáta zostanú v *DataStorage*-i do nasledujúceho plánovaného odovzdávania dát a oznaenie íslom transakcie sa odstráni.)
8. Centrálne môže kedykoľvek po odoslaní konfigurácie zada príkaz na vykonanie „okamžitej“ úlohy. Najčastejšie sa vykonáva ítanie archivovaných dát meraných veličín za obdobie, pre ktoré v centrále chýbajú dáta, ale tiež nastavenie presného času v meraoch a iné. Príkazy sú riadené párovými rozhraniami *I.SGC\_ImmediateReadMaster – Slave*, *I.SGC\_SetRTCMaster – Slave*, *I.SGC\_ConsumerDisconnectMaster – Slave*, kde centrála implementuje *Master* rozhranie a SGCom implementuje *Slave*. Všetky príkazy prebiehajú v režime aplikácie riadených konverzácií ale vždy pozostávajú len z jednoduchej výmeny príkaz – oznámenie výsledku.
9. Ak SGCom stratí spojenie s centrálou, naalej pokračuje vo vykonávaní plánovaných úloh, ako je popísané v bode 7. Zber dát prebieha bez zmeny, všetky pokusy o odovzdanie zlyhávajú a zozbierané dáta sa preto zhromažďujú v *DataStorage*-i až dovtedy, kým sa nepodariť spojenie obnoviť a dáta odovzdať. Po nadviazaní nového spojenia pokračuje vykonávanie bodom 4.

## Modul *TaskExecutor*

Modul *TaskExecutor* je zodpovedný za vykonávanie všetkých úloh, ktoré vyplývajú z konfigurácie ako aj všetkých „okamžitých“ úloh. Používa pri tom zoznam pravidiel, ktoré určujú, ktoré úlohy je možné vykonať paralelne. Úlohám, ktoré paralelne vykonať nemožno (lebo si vzájomne konkurujú v prístupe k meraom), definuje poradie, v akom sa vykonávajú tak, aby bola maximalizovaná priepustnosť. Zároveň tieto pravidlá definujú asové okná, počas ktorých komunikácia s niektorými merami nie je možná – po ukončení spojenia nie je možné určiť as nadviazať nové spojenie.

Pri vytváraní konfigurácie je potrebné brať do úvahy tieto pravidlá, ako aj as, ktorý trvá vytvorenie spojenia, vykonanie jednotlivých úloh, zatvorenie spojenia a výluka spojenia, aby nebolo naplánovaných viac úloh, ako je možné obslúžiť.

Pri obsluhu meraov sa vykonávanie úloh riadi nasledujúcimi pravidlami:

1. Nie je možné vytvoriť súčasne dve alebo viac spojení s jedným logickým zariadením.
2. Nie je možné vytvoriť súčasne spojenia na dve logické zariadenia, ktoré sú pripojené tou istou komunikačnou linkou (v konfigurácii zdieľajú tú istú entitu *Connector*).
3. Ak existuje aktívne spojenie k nejakému logickému zariadeniu, musia sa vykonať všetky úlohy, ktoré sú pre dané zariadenie zaradené do fronty úloh skôr, ako bude spojenie ukonené (z optimalizačných dôvodov). Úlohy sú z fronty vyberané politikou „first-came-first-served“.
4. Po vykonaní poslednej úlohy z fronty zostáva spojenie k logickému zariadeniu aktívne ešte 10 sekúnd Džku akania na novú úlohu je možné upraviť parametrom z príkazového riadku -n.. Ak je v tomto asovom intervale do fronty zaradená nová úloha, je tiež vykonaná a spojenie sa opäť ponecháva aktívne ďalších 10 sekúnd. Až keď nepribudne žiadna nová úloha, je spojenie uzatvorené. (Aby bolo možné efektívne obslúžiť rýchlo sa opakujúce úlohy.)
5. Po zrušení spojenia nie je možné vytvoriť nové spojenie na žiadne z logických zariadení, ktoré sú pripojené použitou komunikačnou linkou po dobu 20 sekúnd Džku ochranného intervalu je možné zmeniť parametrom z príkazového riadku -s.. (Lebo by bolo spojenie aj tak odmietnuté a trvalo by to dlhšie, kým by sa spojenie podarilo opäť nadviazať.)
6. Po uplynutí 20-sekundového ochranného intervalu je obslúžené ďalšie logické zariadenie akajúce vo fronte.
7. Ak je pokus o vytvorenie spojenia s nejakým logickým zariadením neúspešný 3-krát po sebe Počet pokusov je možné zmeniť parametrom z príkazového riadku -r., považuje sa za odpojené – stav *HardError* – a spojenie s ním nie je možné nadviazať po dobu nasledujúcich 5 minút Džku trvania *HardError* stavu je možné upraviť parametrom z príkazového riadku -d.. Všetky úlohy vo fronte, ktoré mali byť vykonané na tomto zariadení, sú stornované, ako aj všetky nové úlohy po dobu trvania *HardError* stavu. Po uplynutí tejto lehoty je možné sa opäť pokúsiť pripojiť k zariadeniu.
8. Všetky úlohy, ktoré si vzájomne nekonkurujú, je možné vykonať paralelne vláknami príslušného *ThreadPool*-u. Počet vláken v *ThreadPool*-e je maximálne 512 Maximálny počet vláken je možné upraviť parametrom z príkazového riadku -t., ale dynamicky sa mení podľa aktuálnych potrieb.