Ako Vyvíjať Driver, Časť 2

V druhej časti lekcie o vývoji Drivera začneme so samotným programovaním a pozrieme sa bližšie na modul Link.

Ako už vieme z prvej časti lekcie, vnútorná architektúra Drivera obsahuje modul Link a modul Packet Processor, ktoré predstavujú komunikačnú vrstvu zabezpečujúcu spojenie Drivera so zariadením. Sú súčasťou SDK dodávaného Gamanetom s názvom Framework 4.

Vo Frameworku 4 sú implementované pravidlá pre Link a Packet Processor, týkajúce sa celkového správania Drivera pri rozpade a obnove komunikácie. Preto ak vývojár použije toto SDK, nemusí sa implementáciou týchto pravidiel zaoberať. Ak sa však rozhodne použiť svoju vlastnú komunikačnú vrstvu, je potrebné, aby tieto pravidlá zabezpečil sám.

Driver Context zabezpečuje zdieľanie spoločných dátových objektov medzi jednotlivými časťami kódu Drivera a Logiky zariadenia.

Inicializácia Logiky zariadenia a Drivera je predgenerovaná v rámci zdrojových kódov vytvorených na Portálových službách v sekcii mySDK.

Plugin Runner, ako hostiteľ inicializovanej instancie Drivera, mu dodá informáciu o ID Bus Controllera, ktorý má obsluhovať. Zároveň je dodaná instancia už inicializovaného Simple Clienta na možnosť komunikácie so Systémom C4, a taktiež aj Cancellation Token pre prípad žiadosti o ukončenie behu Drivera. Vývojár by ho mal používať vo všetkých volaniach tak, aby bolo možné čo najrýchlejšie ukončiť činnosť celého Drivera.

Simple Clientovi je potrebné v rámci inicializácie nadefinovať časovú zónu, v rámci ktorej dané zariadenie pracuje. Štandardne je nadefinovaný UTC čas, ale v Driveri má byť nastavená taká časová zóna, aká je nastavená v rámci jeho parametrov v nastavení Drivera v Systéme C4.

Následne sa v rámci generovaného kódu vytvorí a nainicializuje Driver Context. Systém si z C4 Servera zoberie celý strom, podľa ktorého následne vytvorí celý zoznam instancií objektov, ktoré vývojár pripravil pre jednotlivé typy podsystémov.

Teraz sa bližšie pozrieme na modul Link.

Link je modul Frameworku 4, ktorého hlavnou úlohou je zabezpečiť vytvorenie spojenia so zariadením. Sleduje, či sa spojenie nerozpadlo a ak áno, pokúša sa znova ho nadviazať, a to každých 30 sekúnd.

Link taktiež zabezpečuje aj notifikáciu ostatných častí Drivera, a informuje Systém C4 v prípade zmeny na spojení formou logov a statusov.

Ďalšou úlohou Linku je zabezpečovať fyzické odosielanie dát z Drivera do zariadenia a ich prijímanie spätne v logike daného komunikačného protokolu. Keďže protokoly zariadenia môžu byť rôzneho typu, Link zabezpečuje konverziu informácií spravovaných Driverom do protokolu, ktorému dané zariadenie rozumie.

Keďže Link zabezpečuje komunikáciu so zariadením, musí byť vždy takého typu, aké spojenie umožňuje dané zariadenie.

Vo všeobecnosti môžu byť Linky dvoch kategórií. Prvou sú Linky, ktoré spracúvajú binárny protokol, či už ide o sieťový protokol na báze IP (TCP Client Link, TCP Server Link, UDP Link), alebo fyzické pripojenie zariadenia prostredníctvom RS232 alebo USB pripojenia do počítača.

Druhou kategóriou sú Linky, ktoré zabezpečujú volania vzdialených funkcií, napríklad REST API Link či SQL Link.

Gamanet v rámci SDK dodáva všetky základné typy Linkov, ktoré sa používajú na bežné pripojenie externých zariadení. V prípade špeciálneho typu pripojenia je možné si vyvinúť svoj vlastný typ Linku.

Táto lekcia je zameraná na Linky, ktoré spracúvajú binárne protokoly. V našom prípade budeme vyvíjať Simple Driver na báze TCP klienta.

Vývojár v rámci sekcie mySDK na Portálových službách nadefinuje zoznam Linkov, prostredníctvom ktorých je možné sa pripojiť k danému zariadeniu. Používateľ následne v konfigurácii Systému C4 nastaví príslušný Link podľa typu pripojenia zariadenia do Systému C4 na konkrétnej inštalácii. Na základe tohto nastavenia bude mať vývojár v rámci BusControllerGenerated v property LinkEnum už nadefinovaný konkrétny Link, ktorý si používateľ vybral.

Tento Link sa následne v objekte typu Driver v sekcii OnStart nainicializuje a vloží sa do Packet Processora.

Interná architektúra binárneho Linku sa skladá z niekoľkých podmodulov.

Prvým je Binárny konvertor zabezpečujúci transformáciu dát z vnútorných objektov Drivera typu Paket do binárnej formy, a následne jej odoslanie do zariadenia.

Ďalej obsahuje transportnú vrstvu. Je to dedikovaný thread slúžiaci na asynchrónne odosielanie a prijímanie dát zo zariadenia.

Taktiež obsahuje dva podmoduly, ktoré musia byť vyvinuté vývojárom - Packet Parser a Encryptor.

Packet Parser zabezpečuje spätnú konverziu binárnych dát zo zariadenia do paketov. Ide o špecifický modul, ktorý má v sebe implementovanú Logiku a štruktúru binárnych paketov definovanú v dokumentácii komunikačného protokolu dodaného výrobcom zariadenia. Dáta, ktoré prichádzajú z počítačovej siete, nemusia prísť ako jeden celok. Môžu prichádzať fragmentované - teda po častiach, a to najmä vtedy, keď je zariadenie

vzdialené. Packet Parser funguje ako buffer, do ktorého sa zbierajú bajty a v prípade zozbierania kompletného paketu sa následne transformujú do objektu typu Paket. Tento paket obsahujúci prijaté dáta ďalej prechádza Logikou zariadenia ako jeden objekt.

Encryptor je voliteľný modul, ktorý slúži v prípade enkryptovaného spojenia na enkryptovanie a dekryptovanie dát.

Pozrime sa teraz na to, ako sa Link vytvára a implementuje. Tu vidíme, že sa vytvoril Link a inicializoval sa cez Driver Context. Taktiež tu vidíme ID Bus Controllera a že sa do neho vložil Simple Device Packet Parser. Je to objekt, ktorý slúži na konverziu príchodzích dát do paketov a musí byť naprogramovaný vývojárom.

Na odskúšanie použitia Linku v praxi je potrebné naimplementovať Packet Parser a niekoľko príchodzích a odchodzích paketov.

Rôzne kategórie a typy paketov, ako aj spôsob ich použitia si vysvetlíme v ďalšej časti lekcie.