A Rosetta leszállóegységének szoftver szimulátora

TRÓZNAI GÁBOR, BAKSA ATTILA, SÓDOR BÁLINT

SGF Kft. (Space and Ground Facilities Ltd.)
troznaig@freemail.hu, baksa.attila@syncnet.hu, soba@freemail.hu

Lektorált

Kulcsszavak: Rosetta, leszállóegység, űrkutatás, szoftver, szimulátor, XML, C++, transputer

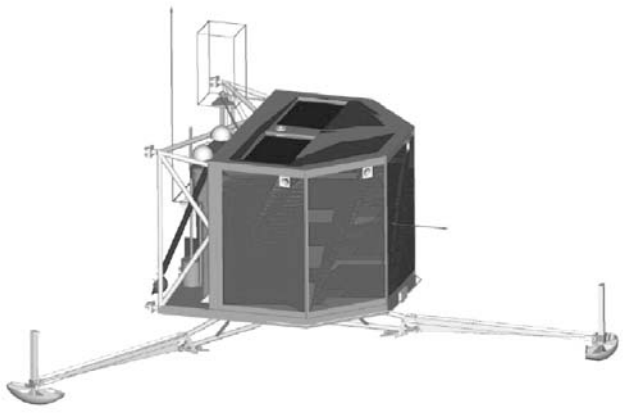
A leszállóegység szoftver szimulátora (LSS) a Rosetta üstökös kutató űrszonda Philae nevű felszíni kutatóegységének földi szimulációját végzi. A szimulátor hardvere öt személyi számítógépből és a gyors válaszidőt biztosító üzenetkezelő kártyákból áll. A leszállóegység berendezéseinek viselkedése egy XML szintaxisú szimulációs nyelv segítségével írható le. Az LSS rendszer tervezésekor a rugalmasság volt a fő szempont. A megvalósított megoldások más hasonló komplex rendszerek működésének szimulációjára is adaptálhatók.

1. Bevezetés

Korábbi Híradástechnika-cikkekben [1,2] már részletesen ismertetésre került a Rosetta leszállóegységének, a Philae-nek a felépítése és feladatai. Jelen cikkünkben csak a szimulátorrendszer bemutatásához szükséges háttérinformációként foglaljuk össze röviden a küldetést. A Rosetta küldetés egy üstökös kutatását tűzi ki célul. A leszállóegység feladata az üstökös felszíni tanulmányozása lesz.

A minél alaposabb mérések elvégzéséhez nyolc tudományos műszert és hét szolgálati alrendszert integráltak a kis méretű kutatóegységbe. A vezérlést egyedi fejlesztésű beágyazott fedélzeti számítógép végzi, amely sokfeladatos operációs rendszerrel és nyolc feladatvégző taszkkal ütemezi a leszállóegység feladatait. A leszállóegység küldetése két fázisra osztható. Az elsődleges küldetés egy rövid, néhány napos ciklus, amikor a leszállás után a lehető legrészletesebb mérések elvégzése a cél, a fő telepek kimerüléséig. Ezt követi a másodlagos küldetés, amelynek során a napelemekre hagyatkozva, alacsonyabb intenzitással, de hónapokon keresztül végzett mérésekkel a Naphoz közelítő üstökösön végbemenő folyamatok elemzése a cél.

1. ábra
A Rosetta leszállóegysége, a Philae



2. Feladatok

A Rosetta űrszonda összetettsége és rendkívül hosszú életútja miatt szükség van egy olyan rendszerre, amely lehetővé teszi a következő feladatok ellátását a Rosetta több mint 10 éves küldetése alatt:

- Kezelőszemélyzet tréningje;
- Üzemeltetési forgatókönyvek ellenőrzése;
- Hosszú időtartamú tesztek;
- Terhelési tesztek;
- Adatforgalmi tesztek;
- Parancs szekvenciák futtatása és tesztelése;
- A fedélzeti számítógép szoftverének tesztelése főként a valódi leszállóegységen kivitelezhetetlen, nem nominális szituációkban;
- Űrszondáról rögzített események reprodukálása.

A fenti feladatok ellátásához az űrszonda földi szimulációjának alapvetően szoftveres úton történő megvalósítása kínálja legmegfelelőbb eszközt. Az SGF Kft. a németországi Deutsche Forschunganstalt für Luft- und Raumfahrt e. V. (DLR) megrendelésére fejlesztette ki a Rosetta Lander Software Simulator-t (LSS-t).

3. Az LSS környezete

A Philae fedélzetén elhelyezett berendezések a leszállóegység központi számítógépével (Command and Data Manegement System – CDMS) állnak kapcsolatban. A CDMS a Rosetta űrszonda fedélzeti számítógépével (On-board Data Handling System – OBDH) tartja a kapcsolatot, az űrszonda elektromos illesztő egységén (Electrical Separation System – ESS) keresztül. A kapcsolattartás két úton történik, az űrutazás alatt az űrszonda és a leszállóegység közötti kábelen, a szétválás után pedig rádió kapcsolat révén valósul meg. Az orbiter viszonylag nagyteljesítményű rádiórendszeren keresztül kommunikál a földi rádióteleszkópokhoz kapcsolódó irányító központtal (Ground Segment), amely a Rosetta űrszonda számítógép szimulátorán át (Space-

craft Interface Simulator – SIS) jut el a Philae irányító központjába (Lander Control Centre System – LCCS). Az adatátvitel a Rosetta Common Packetized Protocol (RPRO) formátuma szerint történik. Az LCCS a Philae fedélzetéről fogadja a tudományos adatokat és kezdeményezi a parancskiadást.

Az LSS szerkezetének tükröznie kell ezt a kommunikációs láncot és a megfelelő szinteken hiteles illesztéseket (interfész) kell biztosítania, amely a következő elemekből áll:

1. Philae fedélzeti berendezések szoftveres szimulációja
2. Philae fedélzeti számítógép (CDMS)
3. Rosetta ESS szoftver szimulációja
4. Rosetta OBDH kommunikációs interfész szimulációja

4. Az LSS felépítése

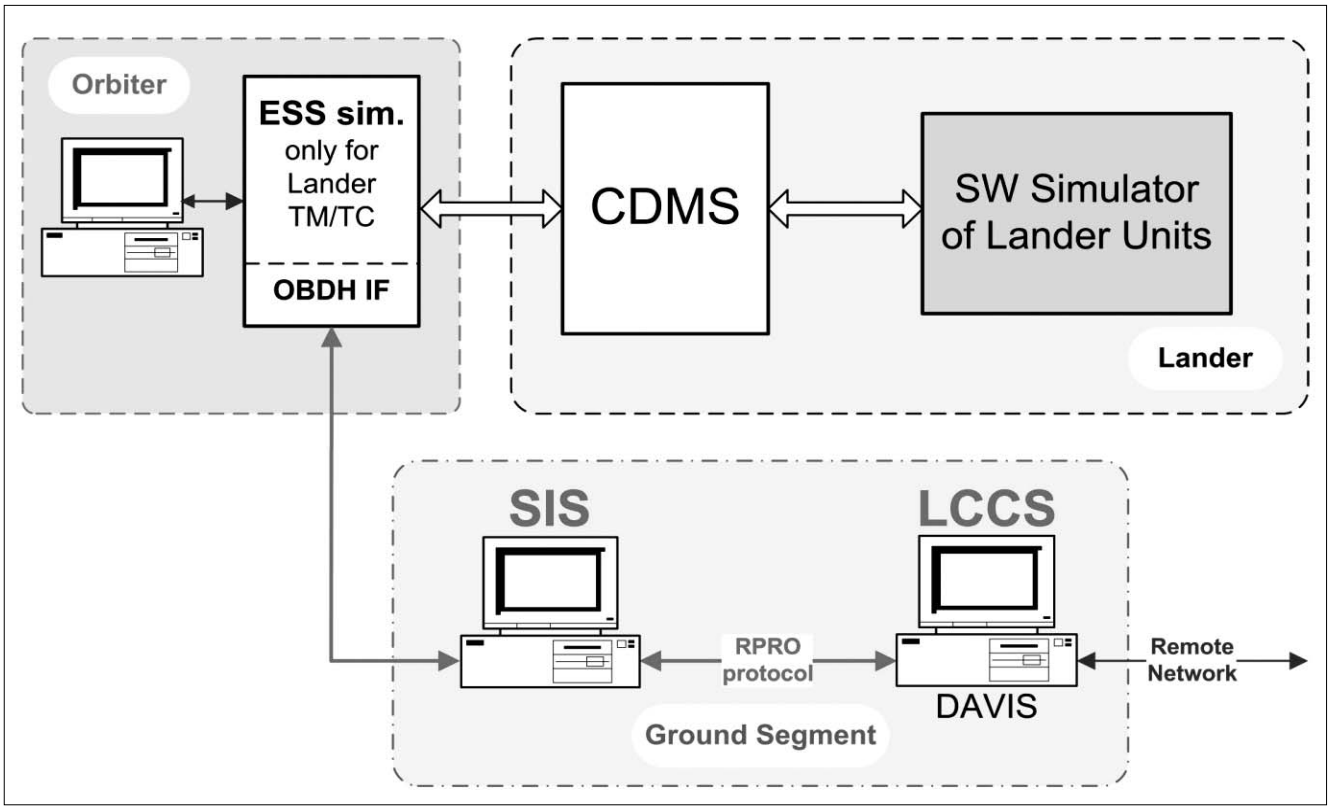
A szoftver szimulátor egy elosztott intelligenciájú, több számítógépből álló hálózat együttese. A különböző berendezések szimulációját négy számítógép végzi, valamint egy ötödik központi számítógép szolgál a szimulációk összefogására és a keletkezett adatok tárolására. A különböző berendezések alacsony szintű, nagy sebességű szimulációját egyedi fejlesztésű hardver elemek, valósidejű üzenetkezelők (Real-Time Message Handlers) végzik, amelyek soros RS-232 portokon kapcsolódnak a számítógépekhez. Az LSS-ben a leszállóegység fedélzeti számítógépét (CDMS) a valósidejű sokfeladatos operációs rendszere miatt, a tényleges reakció idejét, egzakt szimulációját szinte lehetetlen megvalósítani, ezért egy valódi példányt tartalmaz a rendszer. A szimulációs számítógépek Ethernet TCP/IP hálózaton kapcsolatban állnak egymással, valamint a külvilággal.

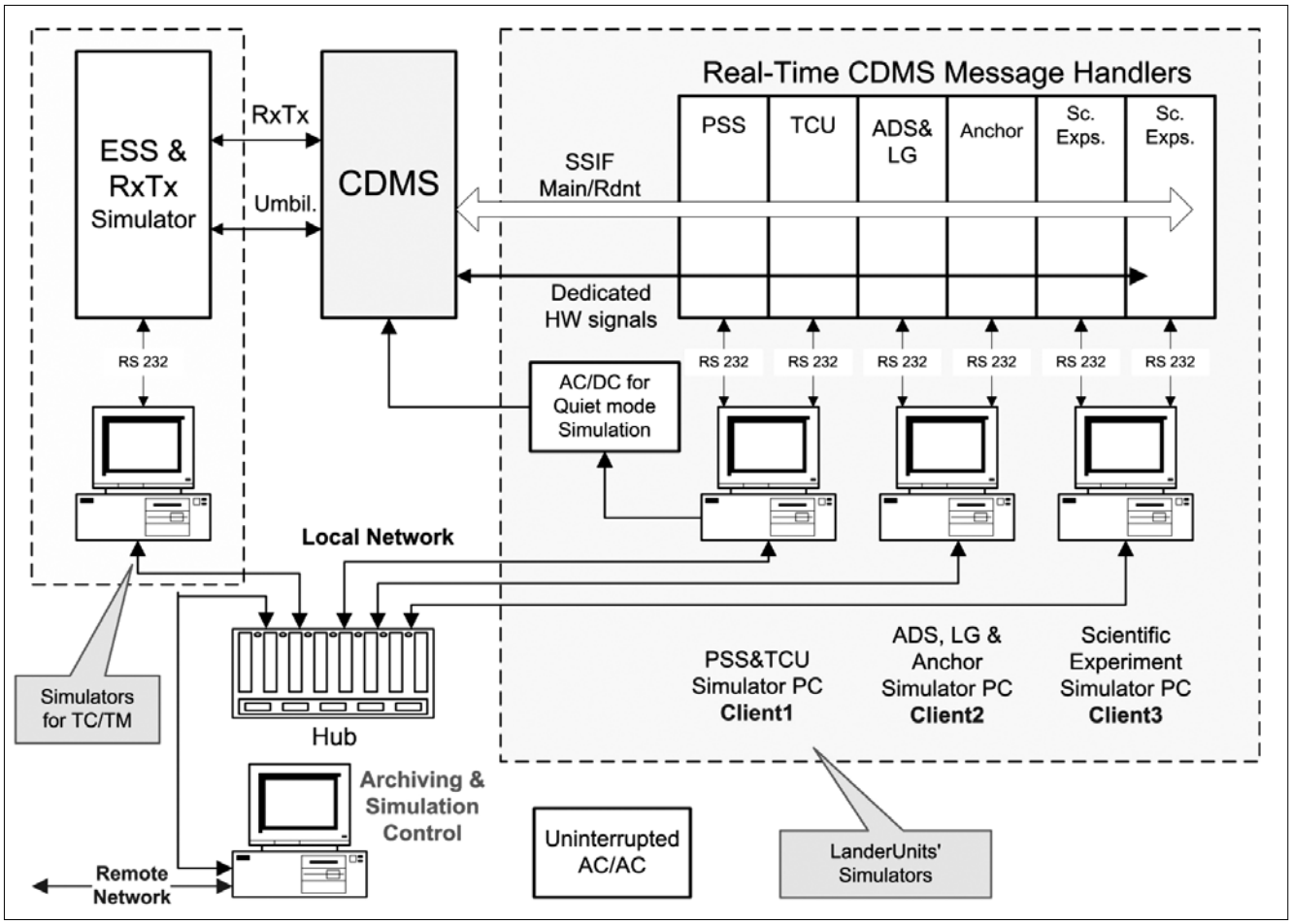
5. Hardver elemek

A Valós Idejű Üzenetkezelő (Real-Time Message Handlers, RIU) kártyák az SGF Kft. által a kilencvenes évek közepén kifejlesztett beágyazott processzort tartalmazó jelszintű szimulátor, amely több célra alkalmazható IBM PC kártya méretű elektronika. A kártya egy transputerre épül, maga a megnevezés a *trans*istor és com*puter* szavak kombinációja, az angliai Inmos cég fejlesztette ki a nyolcvanas évek végén. Egy processzoron belüli párhuzamos processzálásra igen alkalmas architektúrával és az ezt támogató utasítás készlettel, valamint a processzorok összekapcsolását biztosító nagysebességű négy darab soros adatátviteli csatornával rendelkezik mind 16 bites, mind 32 bites processzor változatra. Ez utóbbi tulajdonsága révén nagyszámú processzor összekapcsolását könnyen meg lehetett valósítani. Tulajdonképpen ez a RISC processzor tekinthető a párhuzamos processzálás első igazi megjelenítőjének. Programozása a párhuzamos processzálást igen fejlett szinten támogató OCCAM vagy C nyelven történhet.

Sajnos a megannyi előnyös tulajdonsága ellenére az Intel processzorcsalád tömeges elterjedése halálra ítélte. A beágyazott processzoros szimulátor kártya RS-232

2. ábra A szoftver szimulátor környezete





3. ábra Az LSS felépítése

szabványú soros felületen keresztül csatlakozik a vezérlő és adatfolyam megjelenítő számítógéphez. A kártyán elhelyezett memória mindkét irányú adatforgalom számára átmeneti tárolást biztosít, és lehetővé teszi az előre feltöltött szimulált adatfolyam valósidejű reakcióját.

4. ábra A Valósidejű Üzenetkezelő kártyák

6. Szoftver elemek

A szimulációs rendszer PC-ken futó szoftver elemei két csoportba sorolhatók:

1. A leszállóegység fedélzeti berendezéseinek szimulációja
2. Speciális feladatokat ellátó szoftverek

A leszállóegység fedélzeti berendezéseinek szimulációja

A leszállóegység fedélzeti berendezéseinek szimulációját egy-egy Általános Műszer Modellező modul végzi, a Valós Idejű Üzenetkezelő kártyák segítségével. Ezek a modulok csoportokban is futtathatók, így egy PC-n futó szimulációs szoftver egyszerre több fedélzeti egység szimulációját is végezheti egyidejűleg.

A csoportosítás szabadon változtatható, általában az adott rendszer határozza meg a képzett csoportokat. Több nagy számításigényű szimulációt nem célszerű azonos PC-n futtatni. Ez alapján a jelen rendszerben a következő csoportok lettek kialakítva:

1. PC:
- – Energiaellátó alrendszer (Power SubSystem-PSS)
- – Hőmérséklet Szabályzó alrendszer (Thermal Control Unit-TCU)

2. PC:
- – Leszállást vezérlő alrendszer (Active Descent System-ADS)
- – Leszálló lábak (Landing Gear-LG)
- – Rögzítő horgony (Anchor)
- – Felszíni Mintavevő és fúró rendszer (SD2)

3. PC:
- – Tudományos műszerek (APX, CIVA/ROLIS, CONSERT, COSAC, MUPUS, PTOLEMY, ROMAP, SESAME)

A fedélzeti berendezések viselkedésének leírását egy erre a célra kialakított XML szintaxis alapú szimulációs nyelv teszi lehetővé. Minden tudományos berendezés és szolgálati alrendszer számára önálló szimulációs leírás készíthető, amelyeket az Általános Műszer Modellező modul értelmez és futtat. Minden berendezési modell önálló szálban, saját időrendben és egymástól függetlenül hajtja végre a szimulációs fájlban definiáltakat.

A szimulációs fájl lehetővé teszi a fedélzeti műszerek valós működési üzemmódjainak és az üzemmódok állapot átmeneteinek leírását. A szimulációt végző modulok csoportosítása és paramétereik szintén egy XML alapú konfigurációs fájlban írhatók le. Ezek segítségével a szoftver forráskódjának változtatása nélkül rugalmasan változtatható a szimulációk összeállítása, beleértve azt is például, hogy melyik PC mely fedélzeti egységek szimulációját futtassa. A szimulációs leíró fájlok az XML szintaktikán felül természetesen egy erre a célra kifejlesztett leíró nyelv szintaktikáját is követik, amelyet a szimulátor modul szintaktikai ellenőrzés után értelmez és futtat. Ennek megfelelően, ha egy új egység kerül a rendszerbe, akkor elegendő annak viselkedését a szimulációs leíró nyelven definiálni, amelynek elsajátítása nem igényel komoly fejlesztői ismereteket.

A fejlesztők számára egy további lehetőség új egységeknek a rendszerbe illesztésére egy programozói felület (Application Programming Interface API), amely lehetővé teszi, hogy a rendkívül speciális egységeket – amelyek működése a script nyelven csak bonyolultan írható le – C++ nyelven implementálják, és az API segítségével könnyedén beillesszék a rendszerbe tetemes programozó munkát megtakarítva ezzel. Ez a módszer azonban már komolyabb programozói ismereteket igényel. A jelenlegi szimulációs rendszerben egy ilyen modul fut, az ESS-Bridge (ESS és SIS Simulator). Ez a modul nem használja az általános megközelítésben használatos XML leíró nyelvet. A feladata, hogy modellezze az ESS működését, amely biztosítja a leszálló-egység központi számítógépe (CDMS) és az űrszonda fedélzeti számítógépének földi szimulátora (OBDH és SIS) közötti Kérdés-Válasz jellegű (RTS protocol) kommunikációt mind vezetékes, mind rádió (Rx/Tx) kapcsolaton keresztül.

Speciális feladatokat ellátó rendszerelemek

A második csoportba tartoznak azok a szoftver modulok, amelyek nem berendezések modellezését végzik, hanem az LSS valamilyen speciális feladatát látják el.

LSS Szerver

A szimulációs rendszer TCP/IP szegmensének központi eleme az LSS szerver. A rendszer minden szoftver modulja a szerveren keresztül tartja a kapcsolatot más modulokkal.

A szerver főbb feladatai:
- – kommunikációs kapcsolat biztosítása a rendszer moduljai között,
- – központi adattárolás megvalósítása (Server Data Pool).

Az TCP/IP hálózaton történő kommunikáció egy speciálisan a rendszerre tervezett LSS Data Interchange Protocol (LSDIP) segítségével történik. A protokoll változó méretű adatcsomagokat használ, melyek neve Protocol Control and Data Packet (PCDP).

Ezek egy rögzített méretű fejlécből és egy változó méretű adatrészből állnak. A fejléc tartalmazza többek között a címzett és a feladó modul kódját, azt az információt, hogy a feladó vár-e megerősítést a csomagban kért műveletről, a csomag típusát és altípusát, a csomagra jellemző speciális paramétereket és a csomag adatszegmensének méretét. Az esetlegesen keletkezett átviteli hibák felismerését egy ellenőrző összeg segíti a csomag végén. A modulok a küldeni kívánt adatokat, üzeneteket tehát ilyen PCDP csomagokban továbbítják. A szerver feladatai közé tartozik, hogy kezelje és naplózza a bejelentkezett modulok által nyitott kommunikációs csatornákat és az azokon folyó adatforgalmat.

A Server Data Pool egy központi adatbázis, amely az összes olyan adatot tárolja, amelyekre a moduloknak szükségük lehet a szimuláció során. Ebbe az adatbázisba minden modul szabadon írhat, vagy olvashat PCDP-k segítségével. Az adattartalom változását a szerver nyomon követi és értesítést küldhet azon modulok számára, amelyek változás-figyelési kérést regisztráltak az adott adatterületre.

Az adatbázis szerkezete dinamikusan változtatható akár a szimulációk futása közben is. Az adatbázis szerkezetének kezelését a Simulation Data Pool Presentation/Editor (SDPPE) nevű szoftver végzi.

Simulation Data Pool Presentation/Editor (SDPPE)

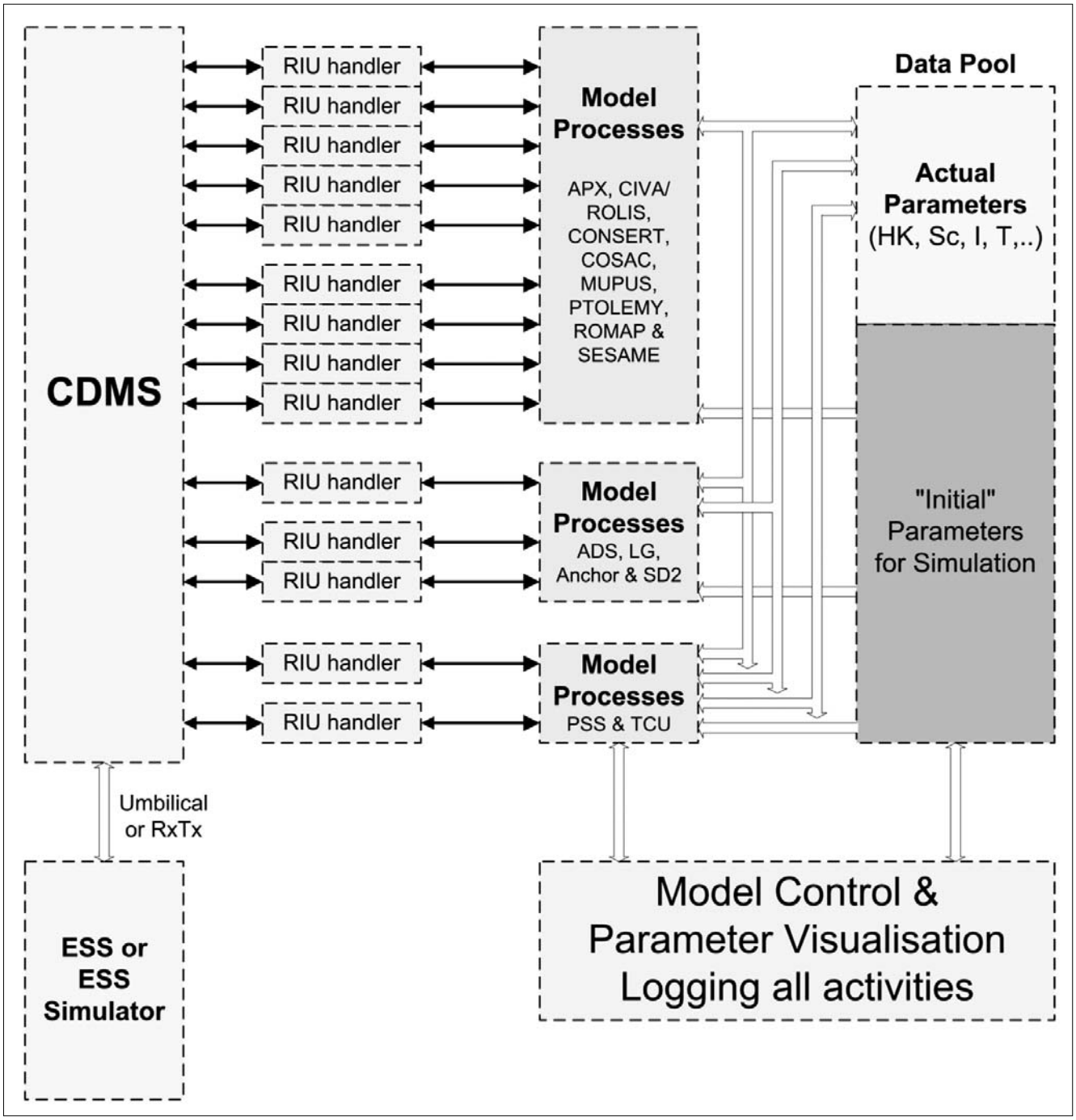
Segítségével egyszerűen össze lehet állítani az adatbázis szerkezetét, meg lehet adni, hogy melyik mező milyen kezdeti értékkel legyen feltöltve, vagy hogy milyen inicializáló fájlból olvassa ki a kezdeti értékeket a program. Ennek megfelelően egy adott pillanatban elmenthető a teljes szimuláció állapota, és egy későbbi újraindítás után ott lehet folytatni a szimulációt, ahol abbamaradt. A Data Pool bármely részébe be lehet tekinteni, és a megfelelő mezőknek manuálisan értéket lehet adni.

A szerver is ad lehetőséget a Data Pool mezőinek megjelenítésére, és folyamatos nyomon követésére, ám az adatok közvetlen editálását ezzel a modullal lehet elvégezni. Ezek mellett a szimuláció vezérlése is megoldható ebből a modulból (leállítás/felfüggesztés/indítás/adatok zárolása stb.). A Data Poolban tárolt adatok egysége a „word” (2 byte). Ezek a szavak nyers (raw) adatok. Általános esetben a raw szó többféle adatot is tárolhat. Például a különböző bitekhez különböző jelentések társulhatnak. Előfordul például, hogy az űreszközön a rendelkezésre álló adatterület maximális kihasználása érdekében például a szó utolsó nyolc bitje egy hőmérséklet értéket tárol, a következő kettő egy 4 állapotú jel értékét, a többi bit pedig 2 állapotú jeleket. Ekkor a hőmérséklet jelet úgy kapjuk, hogy a jelhez rendelt maszkot alkalmazzuk a raw adatra, majd a kapott értéket behelyettesítjük egy a jelhez rendelt matematikai (általában lineáris) kalibrációs egyenletbe, melynek megoldása a valódi hőmérséklet érték. Ennek kódolását és dekódolását több modul is végzi, ahol szükség van a valós adatok megjelenítésére, kiértékelésére, vagy előállítására.

CDMS Memory Tool IF (CMTIF)

A CMTIF feladata, hogy a hozzá érkező kéréseknek megfelelően írási és olvasási műveleteket hajtson vég-

5. ábra A szoftverelemek belső kapcsolatai



re a CDMS memóriájában. Ezt úgy valósítja meg, hogy képes egy Valós Idejű Üzenetkezelő kártyán keresztül közvetlen üzenetváltásra a CDMS belső memóriakezelő moduljával. A kérések érkezhetnek a hálózaton bármely LSS modultól, melyek eredményét a CMTIF visszaküldi a hálózaton a kérést indító felé. Hasonló műveletek elvégzésére lehetőséget ad a program felhasználói felülete is.

CDMS Memory Decoder (LDEME)

A CDMS memóriatartalmának megjelenítését szolgáló kifinomultabb eszköz a CMTIF-fel szorosan együttműködő LDEME (CDMS Memory Decoder). Ez a modul kizárólag TCP/IP kapcsolaton keresztül tartja a kapcsolatot a CMTIF modullal, és a tőle visszakapott adatokat a tartalomnak megfelelően dekódolva jeleníti meg. Így a CDMS memóriatartalma könnyen áttekinthető és értelmezhető. A kommunikáció itt is a szerveren keresztül történik.

Jelenleg ez az egyetlen szituáció, amely igényli a szerverben implementált rugalmas timeout kezelést. A CDMS reakcióideje ugyanis meglehetősen lassú lehet, hiszen fő feladata nem az, hogy kiszolgálja az LDEME és a CMTIF kéréseit. A szerverben megadható ugyan, hogy egy adott modul válaszára mennyi legyen a várakozási idő, ám az LDEME kéréseire adott válaszban szereplő adatmennyiség igen tág határok között mozoghat. Nyilvánvaló, hogy nagyobb adatmennyiség több időt vehet igénybe, így be kellett vezetni egy dinamikus timeout kezelést is a szerverben a fix timeout mellé. Ezzel lehetőség van egyes modulokra a fix timeout érték helyett megadni egy adatmennyiségtől függő timeout értéket. Ekkor a szerver ellenőrzi, hogy a feladó modul mekkora adatot kért a címzettől és ennek megfelelően állítja be arra a csomagra a timeout értékét. Ez a helyzet az LDEME által a CMTIF-től kért adatok esetében is, ugyanis a CMTIF megvárja, még a CDMS megadja a kért választ, és csak ezt követően küldi vissza az LDEME modulnak.

Grafikus adatmegjelenítő (GraphIT)

E modul feladata, hogy grafikus formában, felhasználóbarát módon jelenítse meg a Data Pool aktuális értékeit. Képes ábrázolni az időben változó Data Pool részeket és grafikon formájában, valósidőben rajzolni. A felhasználó összeállíthat különböző grafikonokból csoportokat, melyeket egy ábrában akar kirajzolva látni. Szabadon megadható, hogy a Data Pool melyik részét szeretnénk kirajzoltatni, és milyen formában dekódolni. Vannak ugyanis modulok, melyek lebegőpontos értékeket tárolnak a Data Poolban, így ezek legalább 2 szót foglalnak el, ezért egy ilyen grafikon egy pontjának kirajzolásához mindkét szót le kell kérdezni, dekódolni (esetleg kalibrációs egyenletet alkalmazni rá), majd kirajzolni. A Data Pool tárolhat szöveges adatokat is, amelyek időbeli változását követni tudja ez a modul. Az összeállított grafikon kombinációkat külön ablakokban lehet megjeleníteni, és a teljes konfigurációt fájlba lementeni illetve fájlból visszatölteni.

A grafikonokhoz kétféle frissítési mód rendelhető. Beállítható, hogy a grafikon csak akkor frissüljön, ha a megjelenített adat megváltozott a szerver adatbázisban, vagy periodikusan frissüljön egy beállítható periódus szerint. A megjelenített adatok további feldolgozás céljából fájlba is rögzíthetők, amit aztán más táblázatkezelő vagy adatfeldolgozó programba importálni lehet. Lehetőség van továbbá a Data Pool egy részének kijelölése helyett előre definiált Parameter Object (PO) listából választani. Egy ilyen előre definiált PO, amely például egy hőmérséklet értéket definiál, tartalmazza többek között a hőmérséklet érték alapját képező nyers adat helyét a Data Pool-ban, a dekódolásához szükséges maszkot, és a kalibrálásához szükséges matematikai egyenletet is.

7. Összefoglalás

Az LSS rendszer tervezésekor a rugalmasság volt a fő szempont. Jelenlegi alkalmazása mellett más hasonlóan komplex rendszerek működésének szimulációjára is adaptálható. A moduláris felépítés lehetővé teszi, hogy egyszerre akár sok fejlesztő dolgozzon az egyes modulokon egymástól nagyrészt függetlenül. Egy nemzetközi környezetben folyó hosszú fejlesztés során, mint amilyen a Rosetta program is, ez komoly előnyt jelent.

Az XML alapú leíró nyelv lehetővé teszi különböző berendezések szimulációját, a szoftver forráskódjának változtatása nélkül. A leíró fájlok elkészítése nem igényel mély szoftverfejlesztői tudást a projekt későbbi szakaszába bevont operátoroktól sem. A speciális feladatot ellátó szoftverek nagy része pedig javarészt független attól a konkrét rendszertől, amelynek a szimulációját végezzük. Amennyiben szükséges olyan modul fejlesztése, amely túlmutat az XML leíró nyelv keretein, akkor a fejlesztők munkáját egy C++ API segíti, melynek segítségével tetszőleges új modul a rendszerbe illeszthető.

Irodalom

[1] Baksa Attila:
Űreszközök fedélzeti autonómiájának kialakítása a naprendszer távoli objektumainak kutatásához.
Híradástechnika, 2004/5. sz., pp.30–33.

[2] Dr. Szalai Sándor, Balázs András:
A Rosetta Lander központi vezérlő és adatgyűjtő számítógépe.
Híradástechnika, 2004/5. sz., pp.34–36.