Beágyazott információs rendszerek

Bevezetés (folyt.)

2020. szeptember 10.

Beágyazott rendszerek funkciói

Beágyazott rendszer ~ központi idegrendszer:

→ megfigyel → analizál → dönt → cselekszik

A német gépjármű, automatizálási és orvosi ipar
évente ~15 milliárd € -ot invesztál
beágyazott rendszerek kutatás-fejlesztésére,
miközben éves forgalmuk meghaladja az 500 milliárd € -ot.

Beágyazott rendszerek tulajdonságai:

Intenzív információs kapcsolat
Autonóm működés
Szolgáltatásbiztonság
„Láthatatlanság”

Alternatív elnevezések:

Embedded System
Pervasive Computing
Ubiquitous Computing
Ambient intelligence

Emlékeztető

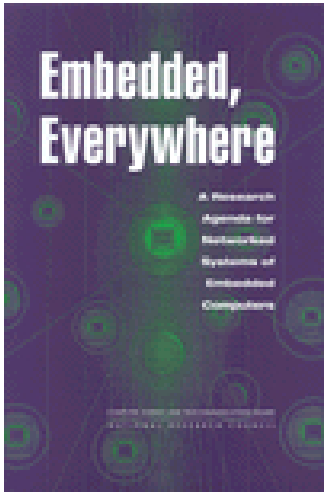
Egy lehetséges definíció:

A befogadó fizikai/kémiai/biológiai környezetükkel intenzív, valós idejű információs kapcsolatban álló,

- emberi beavatkozás nélkül működő,
- nagyon biztonságos,
- sokszor “láthatatlan”

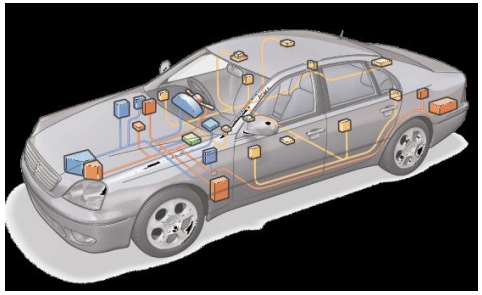
számítógépes rendszerek, melyek

- egy-egy eleme (általában) erősen korlátozott képességű,
- rendszert alkotva azonban (általában) bőséges erőforrásokkal (memória, sávszélesség, ...) rendelkeznek.



A Research Agenda for Networked Systems of Embedded Computers National Academy of Sciences (2001)

Fly-by-wire



Drive-by-wire

BMW 745i:

53 db 8-bites,
11 db 32-bites,
7 db 16-bites processzor,

2 000 000 sor kód,

Windows CE OS,

többszörös hálózat.

A processzorok 2%-a IT és PC felhasználású,
98% beágyazott alkalmazás:
jármű, háztartási gép, mobil telefon, stb.

A főszereplő: a beágyazott szoftver

„Szabványos” hardver és szoftver építőelemek (COTS) alkalmazása mellett,
az egyedi képességeket a beágyazott/alkalmazói szoftver valósítja meg.
A valós rendszerek alkotóelemei egyre inkább „számítástechnikai”
kölcsönhatások révén működnek együtt.
(Prémium kategóriás autók: több ezer jelvezeték, 70 – 100+ elektronikus vezérlő)

A beágyazott szoftver: univerzális rendszerépítő eszköz

Következmények:

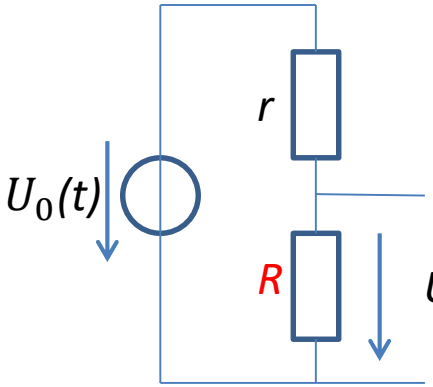
- A szoftver egyrészt abszorbeálja a környezetét,
- másrészt az adott alkalmazás részévé válik.
- A szoftverek a funkcionális és fizikai követelményeknek is eleget tesznek.

„... Software is Hard and Hardware is Soft ...”

Jó hír: szoftverrel megvalósítva sok minden lehetséges …
Rossz hír: szoftverrel megvalósítva sok minden lehetséges …

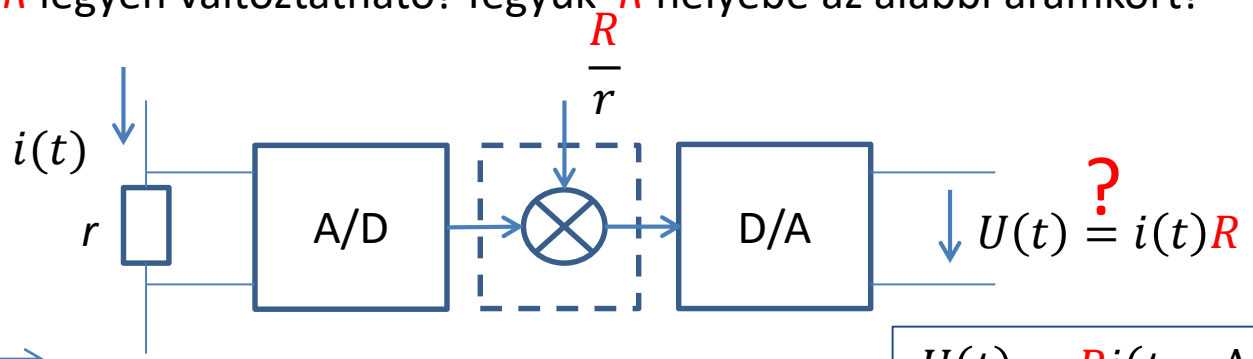
CPS rendszerek modellezési kérdései

Példa: Készítsünk programozható feszültségosztó áramkört-berendezést!

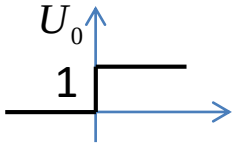


$$U(t) = U_0(t)\frac{R}{r+R} \qquad U(t) = i(t)R \qquad i(t) = \frac{U_0(t)}{r+R}$$

R legyen változtatható! Tegyük R helyébe az alábbi áramkört!



Következmény:



µP, DSP, "felhő", …

$$U(t) = Ri(t-\Delta t)$$

$$U(t=0) = 0 \qquad i(t=0) = \frac{U_0}{r}$$

$$U(t=\Delta t) = R\frac{U_0}{r} \qquad i(t=\Delta t) = \left(U_0 - R\frac{U_0}{r}\right)\frac{1}{r} = \left(1-\frac{R}{r}\right)\frac{U_0}{r}$$

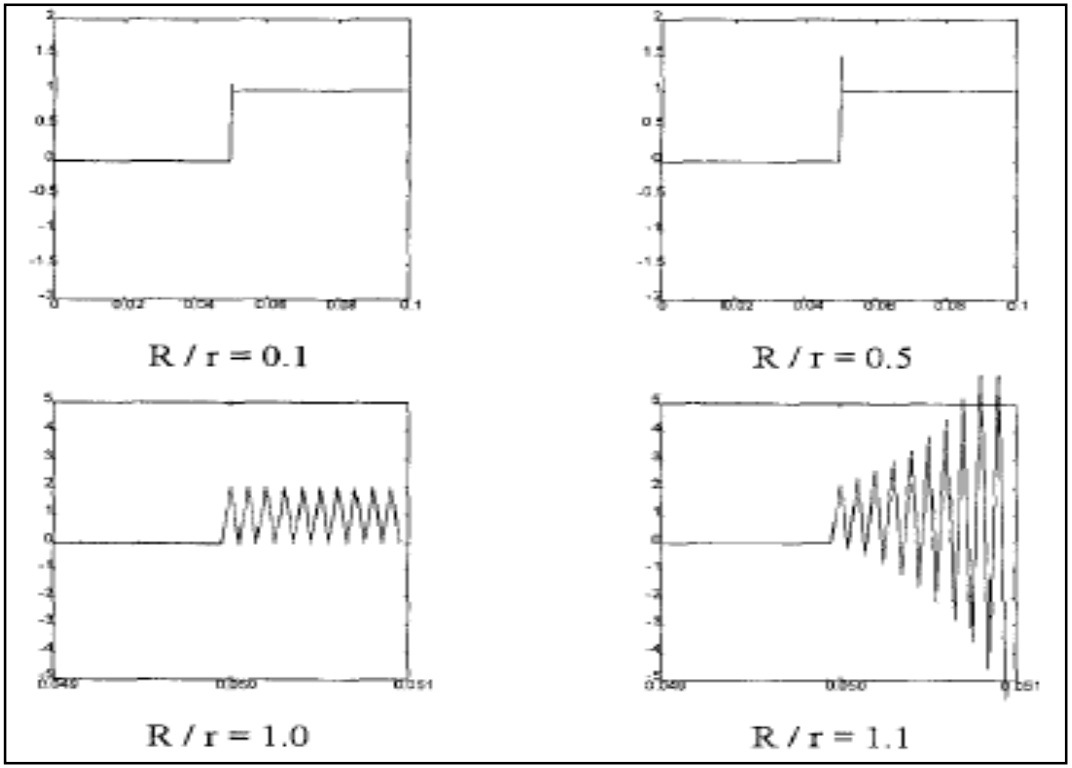
$$U(t=2\Delta t) = R\left(1-\frac{R}{r}\right)\frac{U_0}{r} \qquad i(t=2\Delta t) = \left[U_0 - R\left(1-\frac{R}{r}\right)\frac{U_0}{r}\right]\frac{1}{r} = \left(1-\frac{R}{r}+\left(\frac{R}{r}\right)^2\right)\frac{U_0}{r}$$

CPS rendszerek modellezési kérdései

$$U(t = n\Delta t) = R\left(1 - \frac{R}{r} + \left(\frac{R}{r}\right)^2 \mp \cdots \mp \left(\frac{R}{r}\right)^{n-1}\right)\frac{U_0}{r} \to U_0\frac{R}{r+R}$$

$$i(t = n\Delta t) = \left(1 - \frac{R}{r} + \left(\frac{R}{r}\right)^2 \mp \cdots \pm \left(\frac{R}{r}\right)^{n}\right)\frac{U_0}{r} \to \frac{U_0}{r+R}$$

Ha $\frac{R}{r} < 1$



A késleltetés túllövést,
csillapodó lengést,
állandó amplitúdójú lengést,
növekvő amplitúdójú lengést
okozhat
a paraméterek függvényében!

A folytonos modellek nem alkalmazhatók egy az egyben!

A jövő beágyazott rendszerei: trendek és szóhasználatok

Beágyazott rendszerek (Embedded Systems)
- rendszerek beágyazott szoftverrel …

Hálózatba kapcsolt beágyazott rendszerek (Networked Embedded Systems)
- kommunikáló beágyazott rendszerek …

Rendszerek rendszerei (Systems of Systems)
- kommunikáló és kooperáló rendszerek …

Tárgyak és Szolgáltatások Internete (Internet of Things and Services)
- tárgyak és szolgáltatások kommunikációja és kooperációja …

Kiber-fizikai rendszerek (Cyber-Physical Systems)
- beágyazott rendszerek és a globális hálózatok integrációja

a felhasználó (emberiség) „beágyazása” érdekében!

Cél az új minőség:

mindenki életvitelében,
az egészségügyi ellátásban, az élelmiszer termelésben és ellátásban,
az idősekről és az elesettekről történő gondoskodásban,

és mindezek érdekében
az energiagazdálkodásban, a közlekedésben, a környezetvédelemben,
a katasztrófák elleni védelemben, az élet- és vagyonvédelemben, …

Európai kezdeményezések:

FP5, FP6, FP7 programok, Eureka ITEA, ARTEMIS: Advanced Research & Technology for Embedded Intelligent Systems, Horizon 2020 előkészítés, CHIST-ERA, Alliance for Internet of Things Innovation (AIOTI), Industry 4.0, ...

Kiemelt alkalmazási területek:

- Hatékony és biztonságos mobilitás (szárazföldi és légi, …)
- Jólét és egészség (otthoni-kórházi ápolás, …)
- Fenntartható termelés (élelmiszer, energia, bányászat, …)
- Intelligens közösségek (intelligens és biztonságos városok, terek, …)

A kihívások és lehetőségek címszavai: biztonságkritikus rendszerek, virtuális világ, nagymennyiségű adat, felhő szolgáltatások, rendszerek rendszerei, autonóm, adaptív és prediktív szabályozás, tárgyak internete, számítások sokmagú processzorral.

+ Horizon 2020: Leadership in enabling and industrial technologies

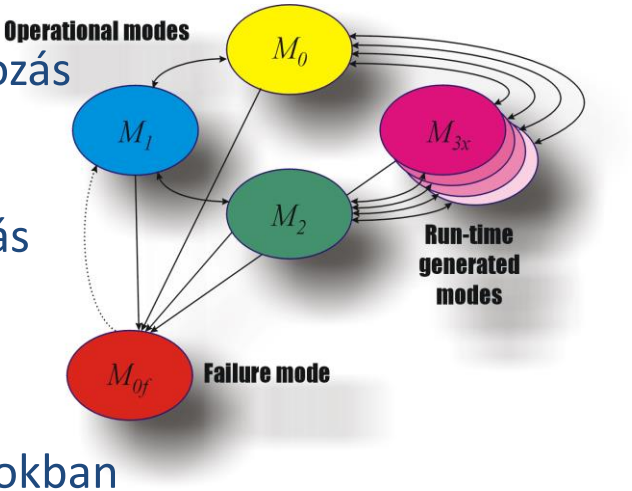
Smart Cyber-Physical Systems ICT-01-2014, ICT1.1-2016
Smart System Integration ICT-02-2014, ICT1.3-2016
Smart Anything Everywhere Initiative ICT1.4-2016
IoT and Platforms for Connected Smart Objects ICT-30-2015
R&I on IoT integration and platforms ICT7.3 – 2016
Smart Anything Everywhere Initiative ICT-04-2017

Kihívások, feladatok, további megalapozó kutatások-fejlesztések

Az adat- és jelfeldolgozás területén:

A valós idejű adat minősége és a kapcsolódó feldolgozás lehetőségei

- Adat pontosság/érvényesség/elévülés, adatvesztés
- Nem egyenletes mintavételezés, órák és adatok szinkronizációja
- Kvantálási hibák időben és amplitúdóban
- Modellillesztés, modell-alapú és adaptív jelfeldolgozás



A rendszer- és irányításelmélet területén:

A többszintű és elosztott rendszerek irányítása

- Hálózatba kapcsolt rendszerek stabilitása, passzivitás alapú rendszerek
- Adaptivitás és kooperativitás: átkapcsolás és újrakonfigurálás, tranziens menedzsment
- Hibrid rendszerek, hibrid szimuláció: hardver-a-hurokban
- Robusztusság, szolgáltatásbiztonság, hibatűrés

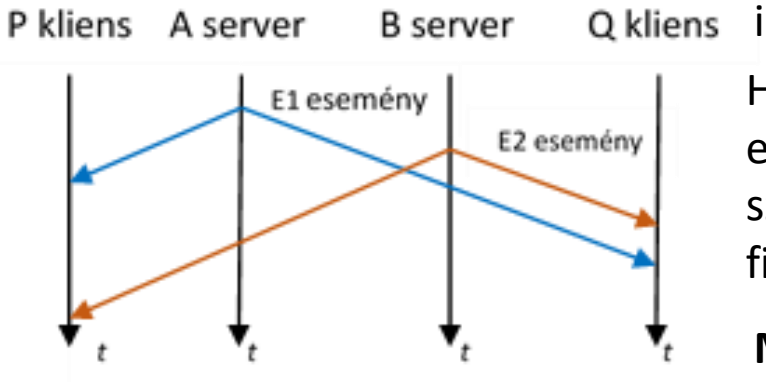
A szoftver rendszertechnika területén:

- Modell-alapú rendszertervezés
- Beágyazott virtualizáció, beágyazott rendszerek felhőben

+ a fejlesztési technológiákhoz, rendszer és hálózati szoftverekhez, a verifikációs, validációs és tanúsítási eszközökhöz kötődő szerteágazó K+F+I

Példák az időviszonyok sajátosságaira beágyazott rendszerekben:

- **relativisztikus hatás**: a kommunikáció időviszonyai események tényleges sorrendjét a vétel helyén megváltoztathatják. Az ábrán az látható, hogy a **Q** kliens esetében az **E2** eseményről szóló híradás megelőzi az időben korábbi **E1** eseményről érkező híradást.



Ha az **E1** és az **E2** események nem függetlenek egymástól, akkor jogos felvetés, hogy az **E2** eseményről szóló híradás megérkezését követően a híradásokat figyelembe vevő döntéseinkkel várakozzunk.

Meddig?

Addig, amíg minden olyan eseményről szóló híradás, amely az **E2** eseménnyel egyidejűleg vagy azt megelőzően történt – a legkedvezőtlenebb esetben is – beérkezik a **Q** klienshez.

Ezt a várakozási időt **akció késleltetési időnek** (*action delay*) nevezzük.

A szükséges akció késleltetési időt akkor tudjuk meghatározni, ha van minimális (alsó) és maximális (felső) korlátunk az üzenettovábbítási időre, azaz a d üzenettovábbítási időre fennáll:

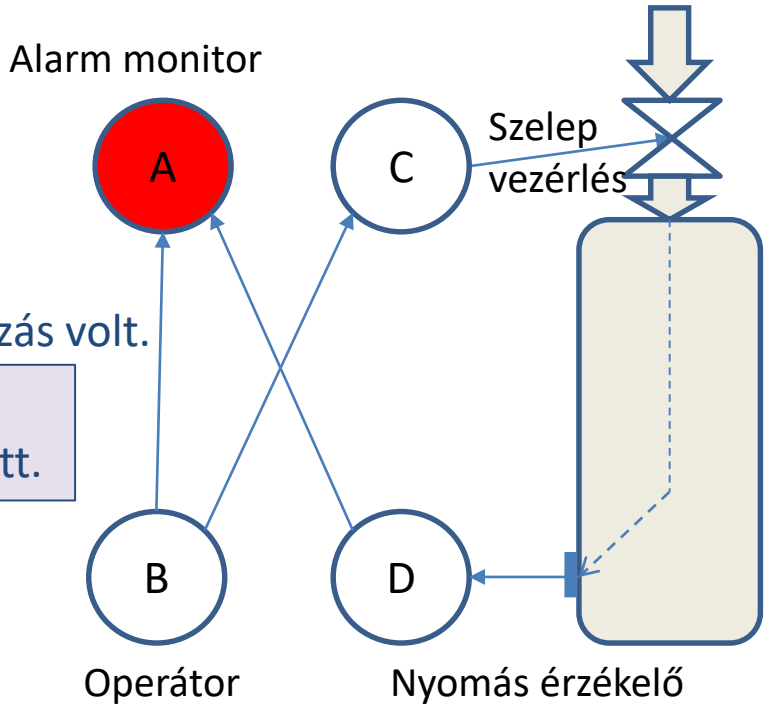
$$d_{min} \leq d \leq d_{max}$$

Mennyiségek, változók valós idejű rendszerekben

Példa: Egy tartályban lévő nyomást monitorozunk egy elosztott rendszerrel.

A csomópont: alarm monitor,
B csomópont: operátor,
C csomópont: szelep vezérlés,
D csomópont: nyomás érzékelő.

Lehetséges üzenetek:
$\mathbf{M}_{DA}$: jelzi, hogy a nyomás hirtelen megváltozott,
$\mathbf{M}_{BC}$: operátori parancs a változtatásra,
$\mathbf{M}_{BA}$: nincs alarm helyzet, mert operátori beavatkozás volt.



Van egy eltakart, a fizikai rendszer működéséből adódó csatorna a szelep és a nyomásérzékelő között.

Téves riasztás jöhet létre, ha
a $B \rightarrow C \rightarrow D \rightarrow A$ láncon gyorsabban fut végig az információ, mint a $B \rightarrow A$ láncon.

Ennek elkerülése érdekében az alarm monitor minden akcióját késleltetni kell.
(Bizonyos akciók visszavonhatatlanok: pilóta katapultál, lőfegyver elsül, stb.)

Vegyük észre, hogy maga a technológia Is kommunikációs csatornát valósít meg!

Megjegyzés: Párhuzam az 1986-os csernobili katasztrófával!

Mennyiségek, változók valós idejű rendszerekben

Akció késleltetési idő (action delay): amíg érvényessé nem válik az üzenet. (Ezt mindig ki kell várni.) Számítása, ha (1) van globális óra:

$t_{érvényes} = t_{küld} + d_{max} + 2g$, ahol g az óra felbontása. Számítása, ha (2) nincs globális óra:

$t_{érvényes} = t_{küld} + 2d_{max} - d_{min} + g_l$, ahol g_l a lokális óra felbontása.

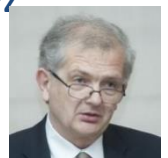
Látható, hogy a második esetben $d_{max} - d_{min}$ idővel többet kell várni, mert valójában a küldés ideje nem ismert, míg az első esetben a küldés időpontja az üzenet részeként elküldhető. Nagy $d_{max} - d_{min}$ érték esetén, ilyenkor lényegesen kedvezőtlenebb helyzettel állunk szemben. Célszerű az üzenetküldési időt ~állandó értéken tartani.

Bizonyos kommunikációs protokollok esetén azonban a $d_{max} - d_{min}$ különbség nagy. Például tokenvezérelt busz esetén, ha a token körüljárási idő mondjuk 10 *ms*, maga az üzenettovábbítás pedig mindig 1 *ms*, akkor $d_{max} = 11ms$ lesz, míg $d_{min} = 1$ *ms*, hiszen a legkedvezőtlenebb esetben az üzenettovábbítás kezdeményezését közvetlenül megelőzi a token továbbítása az adott csomópontról/készülékről, tehát 10 *ms*-ig várni kell.

További megjegyzések:

(1) Az akció késleltetési idő számítására vonatkozó gondolatmenet megértését segíti, ha elképzelünk egy külső megfigyelőt, aki minden időpontot ismer, és tisztában van azzal is, hogy az egyes csomópontokban mi ismert és mi nem.

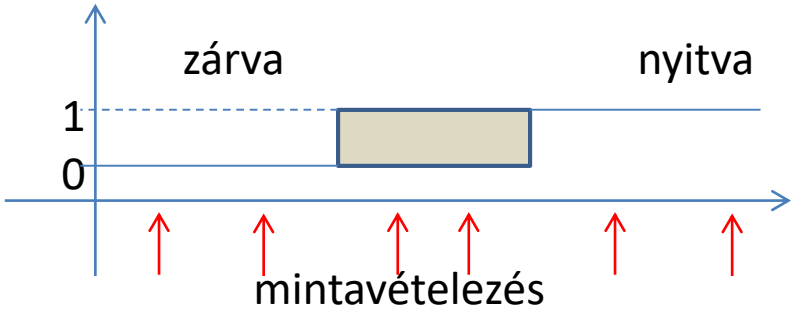
(2) Egy RT kép csak az **állandóság** (permanence) bekövetkezése után használható. Ha ez nagyobb, mint az RT kép időbeni pontossága, akkor csak az állapotbecslés segíthet.



Mennyiségek, változók valós idejű rendszerekben

Állandóság (Permanence). Jelentése: megmarad/stabilizálódik/érvényessé válik az üzenet állapota. Egy üzenet akkor válik állandóvá/megmaradóvá/érvényessé, amikor a vevő csomópont tudja, hogy minden, a jelen üzenet küldési ideje előtt elküldött üzenet már meg kellett érkezzen, vagy sosem fog megérkezni.

- ***Real-time változók*** (RT entities): állapotváltozók, mint pl. folyadék áram, szabályozó alapjele, szabályozó szelep kívánt pozíciója.
  Vannak statikus és időben változó, dinamikus attribútumai.
- ***Befolyásolhatósági tartomány*** (sphere of control, SOC):
  Minden *RT változó* annak az alrendszernek az ún. befolyásolhatósági tartományában van, amelyik jogosult értékét megváltoztatni.
  Azon kívülről a *RT változó* csak olvasható.
- Egy *RT változó* lehet diszkrét vagy folytonos értékű.
- A diszkrét *RT változó* lehet definiálatlan.
  Példa: nyíló garázsajtó: nincs se nyitva, se csukva.



Mennyiségek, változók valós idejű rendszerekben

- ***Megfigyelések:*** a RT változó értékei adott időpont(ok)ban.

Megfigyelés =<név, megfigyelési idő, érték>

- *Megfigyelések elosztott rendszerekben:*
  Ha nincs globális óra, akkor az időbélyeg használhatósága korlátozott, megfigyelési időnek sokszor az üzenet érkezési idejét veszik.
  Ezzel jelentős hibát okozhatunk az állapotbecslésben.
- *Indirekt megfigyelések:*
  Sokszor a megfigyelendő mennyiség közvetlenül nem férhető hozzá.
  Ilyenkor közvetett megfigyeléseket végzünk modellek felhasználásával.
  (Például belső hőmérséklet megfigyelése a felszínen elhelyezett érzékelőkkel).
- *Állapot megfigyelések:*
  Minden megfigyelés önállóan értelmezhető értéket ad.
  Jellegzetesen periodikus mintavételezéssel végezzük.
- *Esemény megfigyelések:*
  Az esemény adott időpontban bekövetkező állapotváltozás.
  Mivel maga a megfigyelés is egy esemény, ezért nem lehetséges egy esemény közvetlen megfigyelése az irányított objektumban, csak annak következményeit tudjuk megfigyelni.

Mennyiségek, változók valós idejű rendszerekben

- ***Real-time változók képe*** (RT image):

  a RT változó megfeleltetése a számítógépes programban, amelynek értelmezzük az *időbeni* és az *amplitúdó szerinti pontosságát*, valamint az *időbeni érvényességét*.

  Egy ***RT változó képe***:
  aktuális állapot, ill. esemény megfigyelés, vagy állapot becslés.

- ***Real-time objektumok*** (RT objects):

  Egy RT objektum az elosztott rendszer csomópontján belül egy olyan **tároló**, amely egy **RT változót**, vagy annak **képét** tartalmazza.

  Minden ilyen objektumhoz tartozik egy **előírt pontosságú óra**.

  Amikor ez üt, egy objektum eljárás aktiválására kerül sor.

  Ha ez periodikus, akkor **szinkron RT objektumról** beszélünk.

  Elosztott RT objektumról beszélünk, ha a különféle csomópontokban **másolat** formájában van jelen.
  Erre jó példa a **globális óra**, amelynek Π együttfutású **másolatait** hozzuk létre az egyes csomópontokban.

- ***Időbeni pontosság:***

  A megfigyelések révén szerzett információ időbeni megjelenése a számítógépes programban és tényleges megfigyelés tényleges időpontja óhatatlanul eltérnek egymástól.

  Az időbeni pontosság azzal a $d_{pontosság}$ intervallummal definiálódik, amelyhez tartozóan bekövetkező amplitúdó hiba még éppen elviselhető a vezérelt rendszer szempontjából.

Mennyiségek, változók valós idejű rendszerekben

- ***Példa:*** az alábbi táblázatban néhány gépjármű motor jellemző szerepel együtt a megkívánt amplitúdó pontossággal és az ennek megfeleltethető időintervallumokkal.

RT kép a számítógépben	max. változás	pontosság	időbeni pontosság
Dugattyú pozíció	6000 ford/perc	0.1°	3μsec
Gázpedál pozíció	100%/sec	1%	10 msec
Motor terhelés	50%/sec	1%	20 msec
Olaj és hűtővíz hőmérséklet	10%/perc	1%	6 sec

A RT képek pontossági intervallumai között több, mint 6 nagyságrend eltérés van!
A dugattyú pozíció esetében ez a pontosság praktikusan csak állapotbecsléssel (a programon belüli jóslással) lehetséges.

A megfigyelés és a felhasználás között eltelt idő egy *v* változó esetén a következő hibát okozza:

$$hiba(t) \cong \frac{dv(t)}{dt}\left[C(t_{felhasználás}) - C(t_{megfigyelés})\right]$$

Ha egy időben pontos RT képet használunk, akkor a *worst-case* hiba:

$$hiba = \underbrace{\max_{\forall t}}\left|\frac{dv(t)}{dt}\right| d_{pontosság}$$