Magyar Tudományos Akadémia
Számítástechnikai és Automatizálási Kutató Intézete

Ellenőrzőpont támogatás PVM alkalmazások számára a magyar ClusterGriden

Kovács József,

Farkas Zoltán, Marosi Attila Csaba

Laboratory of Parallel and Distributed Systems

MTA SZTAKI

{smith,zfarkas,atisu}@sztaki.hu

www.lpds.sztaki.hu

Miről is lesz szó?

- ***Mi az az ellenőrzőpontozás (checkpoint)?***
- ***Klasztergrid***
- ***Checkpoint struktúra***
- ***Fontosabb megvalósítási technikák***
- ***Főbb protokollok***
- ***Összegzés***

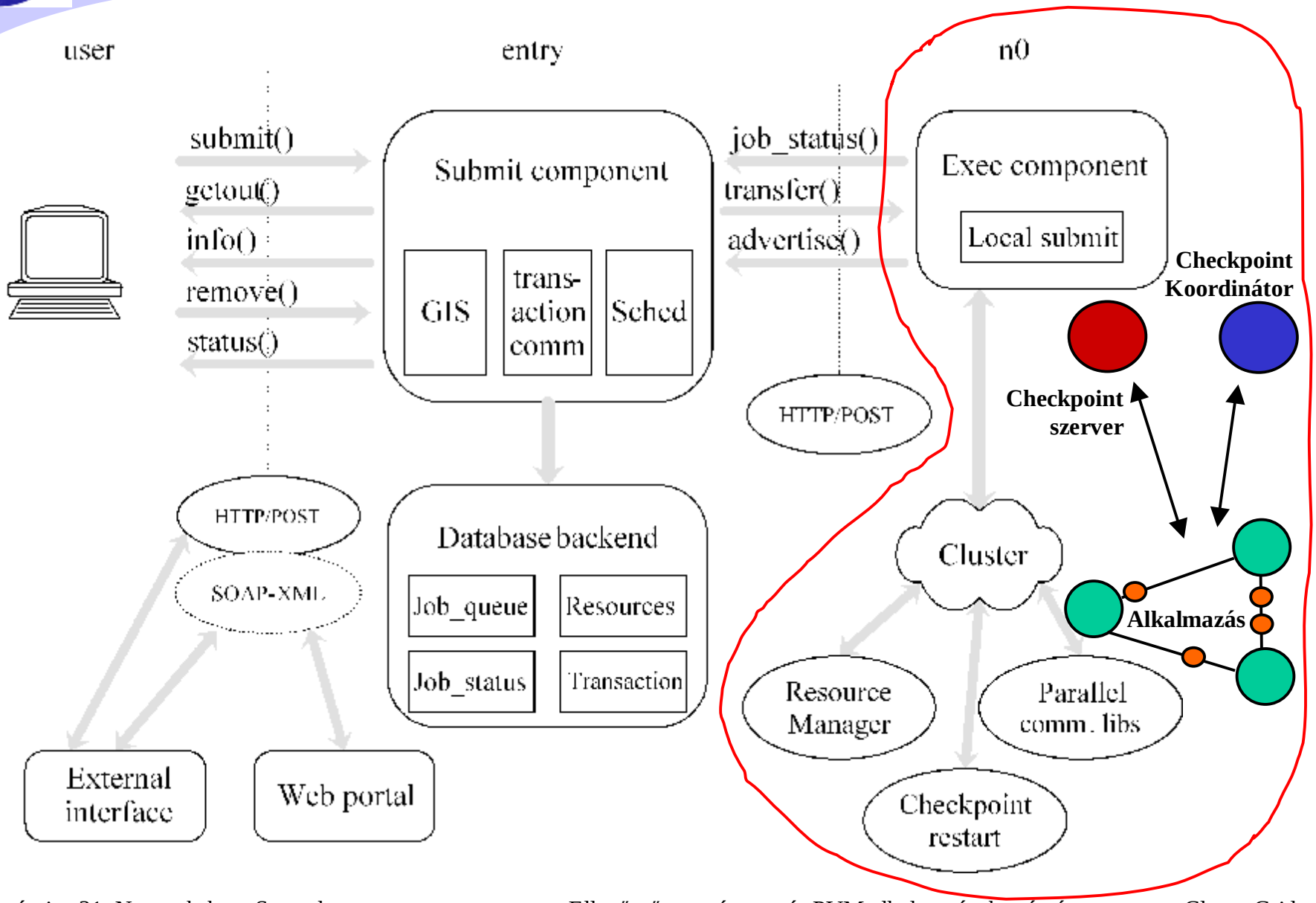
Mi a Checkpointing?

Egy futó alkalmazás teljes állapotterének lementése oly módon, hogy az a lementési pontból újraindítható (folytatható) legyen

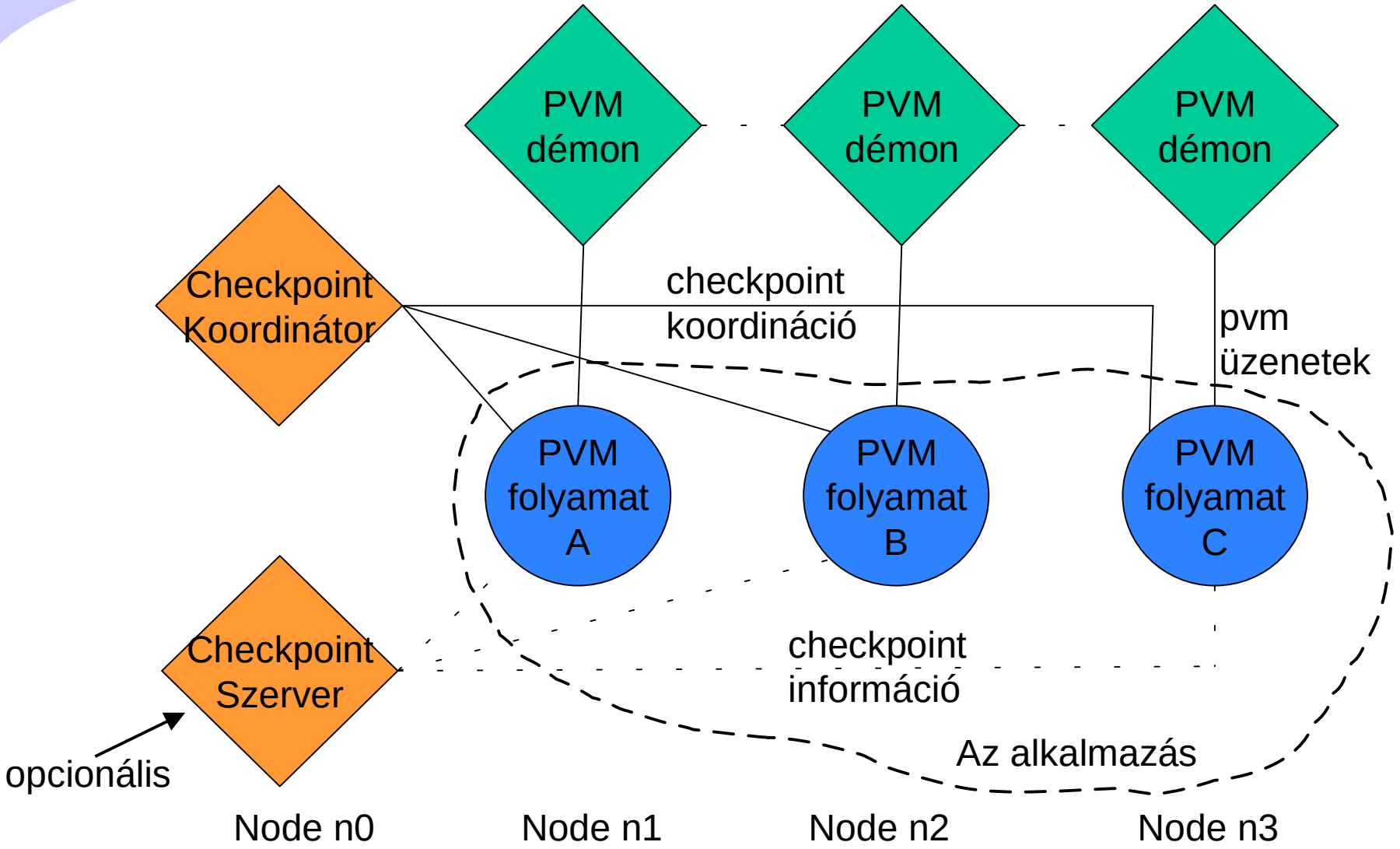
A Checkpoint (és migrációs) támogatás szükséges

- A terheléselosztás megvalósításához (load-balancing)
  - Túlterhelt gépekről folyamatokat migrálunk a kevésbé terheltekre
- A nagyáteresztésű ütemezés megvalósításához (high-throuput computing)
  - A szabad erőforrások kihasználása
- Hibatűrés megvalósításához: node-ok leállhatnak (fault-tolerant)
  - Hardware ill. software meghibásodások miatt
  - Hálózati meghibásodás miatt
  - Adminisztrációs okokból (reconfiguration, upgrade)
- Speciális erőforrás igények kielégítéséhez
  - Egy folyamat akár migrálható egy speciális erőforrás meglétének helyére

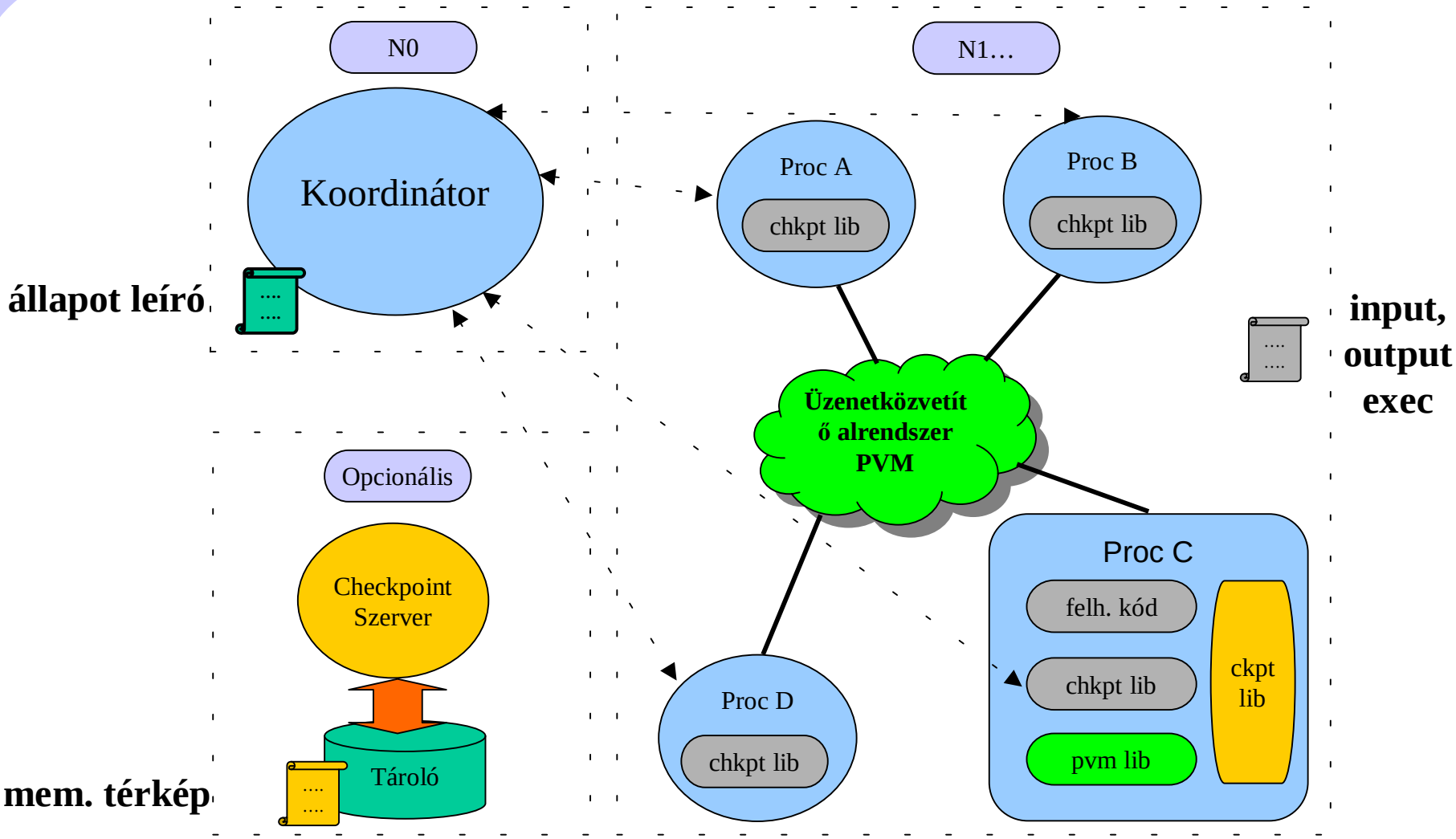
A KlaszterGrid infrastruktúra és checkpoint kiegészítés



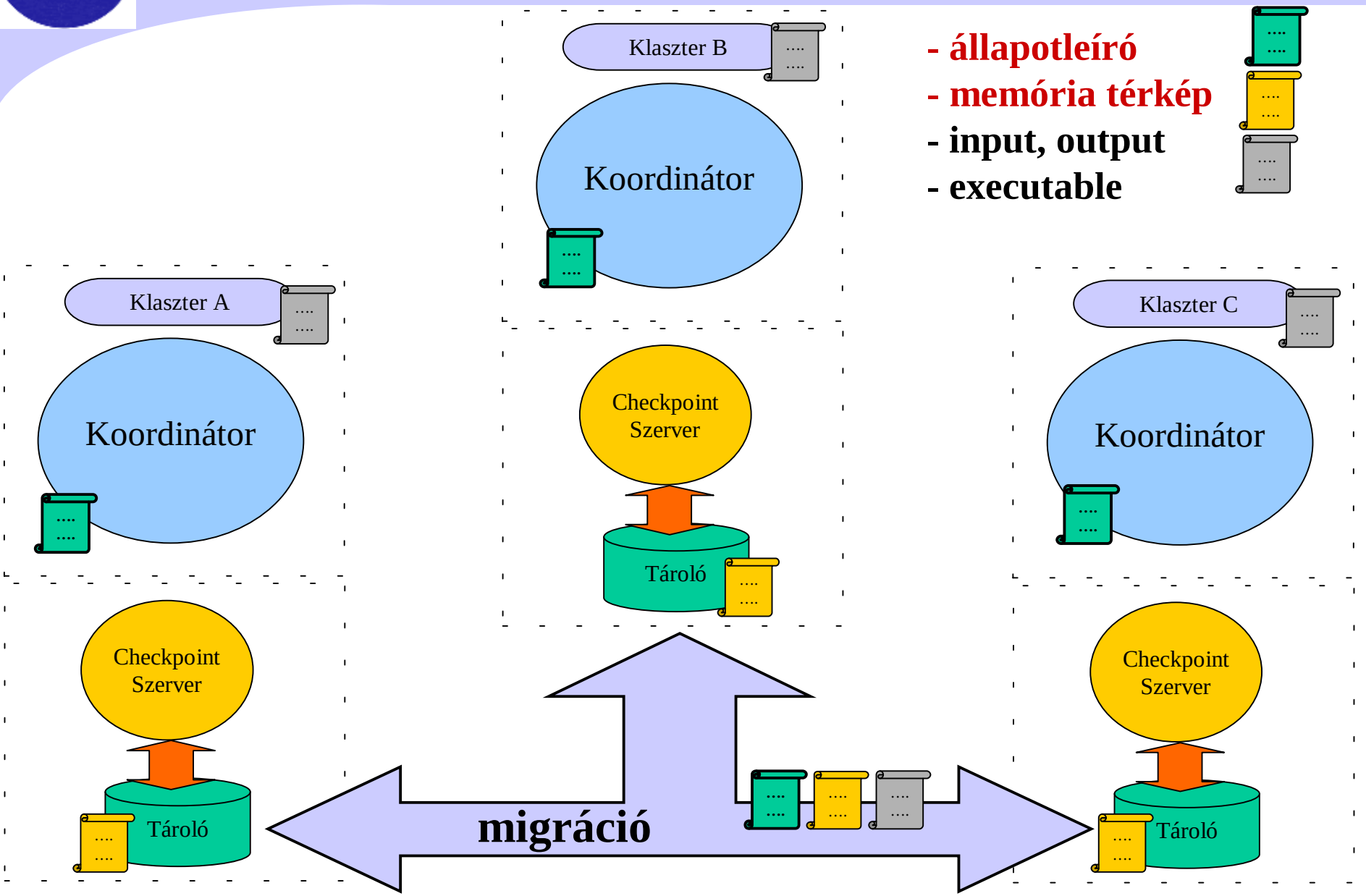
A checkpoint-oló rendszer felépítése



Az alkalmazás és környezete felépítése

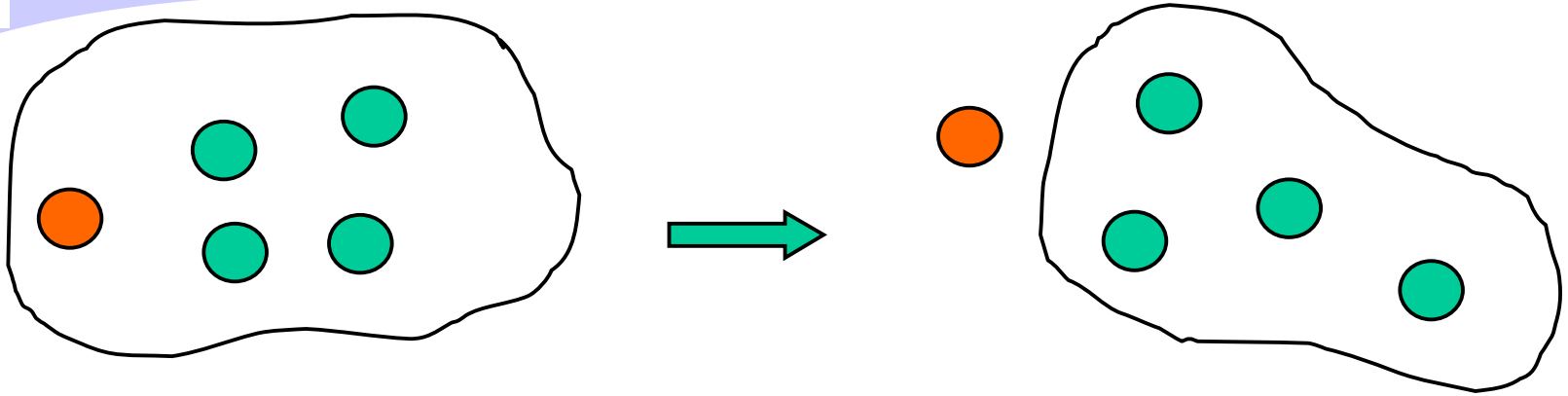


A migráció alapvető alkatrészei



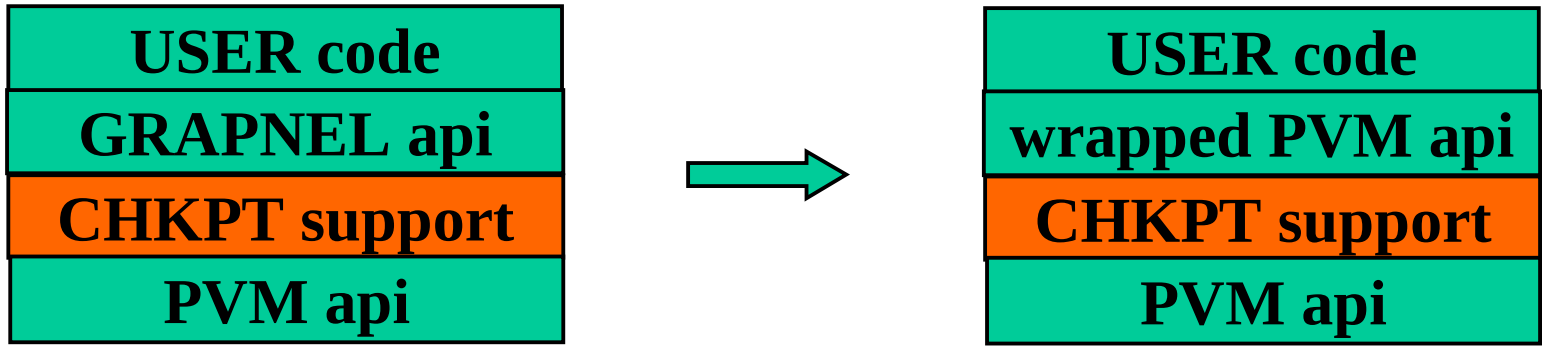
- állapotleíró
- memória térkép
- input, output
- executable

Összehasonlítás a korábbi GRAPNEL checkpointolással



A koordinációs folyamat **része** az alkalmazásnak

A koordinációs folyamat egy **különálló** démon



az üzenetek formátuma és a topológia lekérhető a **GRAPNEL rétegből**

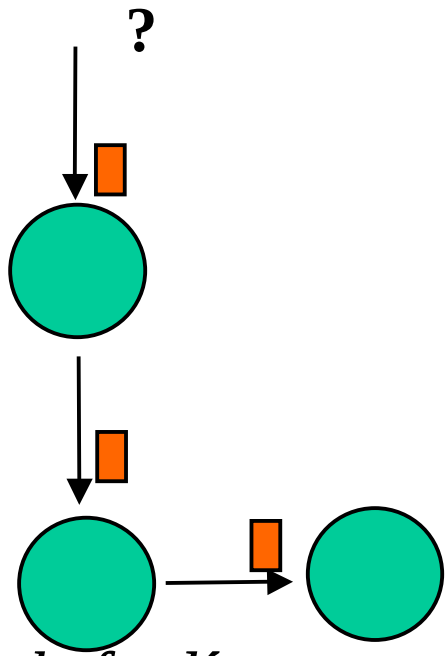
az üzenetek formátuma és a topológia teljes mértékben **elrejtve**

A checkpointoláshoz szükséges mechanizmusok

- A checkpoint támogatás aktiválása/deaktiválása
- Környezeti beállítások eljuttatása a checkpoint réteghez
- Üzemmódok beállítása
- Pvm azonosítók konzisztenciája
- Megszakíthatóság
- Úton levő üzenetek kezelése
- Üzenetek és üzenetbufferek kezelése
- Üzenetközvetítő réteg kezelése (kapcsolat építés, bontás)

Módosítások I.

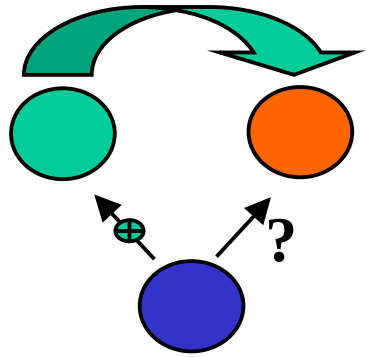
- *aktiválás/dektiválás*
  - *Checkpointoljunk vagy ne?*
- *környezeti beállítások*
  - *Hol van a koordinációs folyamat?*
- *üzemmódok*
  - *Alkalmazás indításakor kérünk visszaállítást vagy nem?*



•A checkpoint támogatás a végrehajtandó fájlba van befordítva
•Az inicializálás minden újonnan létrejött folyamat elején lefut
•A vezérlés környezeti változók értékein keresztül történik
•Ezen értékeket a gyermek folyamatoknál is be kell állítani
•A folyamat létrehozó függvény pvm_spawn() módosítása oly módon, hogy ezek a paraméterek öröklődjenek

Módosítások II.

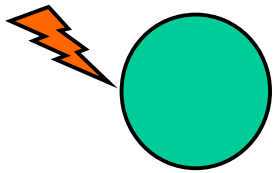
- *Azonosító konzisztencia*
  - *Azonosítók a folyamatokhoz és bufferekhez rendelve*
  - *PVM démonok nincsenek lementve, egyszerűen újraindítjuk őket*
  - *A le, majd felkapcsolódó folyamatok és a bufferek azonosítói megváltoznak*
  - *A felhasználói kód tárolhatja és hivatkozhat rájuk*

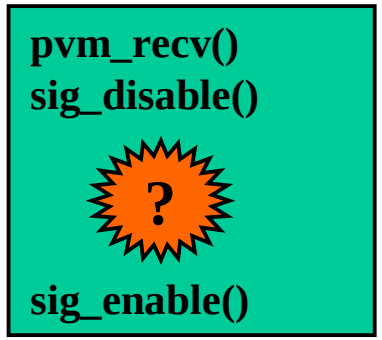


•Az ütközések elkerülése érdekében a rendszer és a felhasználó által tárolt azonosítók között egy összerendelés valósul meg
•A koordinátor terjeszti az új azonosítókat
•Minden új folyamat a koordinátornál jelentkezik azonosítójával
•Minden új folyamat lekéri a létező folyamatok azonosítóit

Módosítások III.

- *Megszakíthatóság*
  - *Aszinkron checkpointot kell támogatni*
  - *Nem reentráns pvm hívásokat atomivá kell tenni*
  - *Biztosítani kell, hogy a blokkolt, atomi műveletek is “rövid” időn belűl befejeződik*

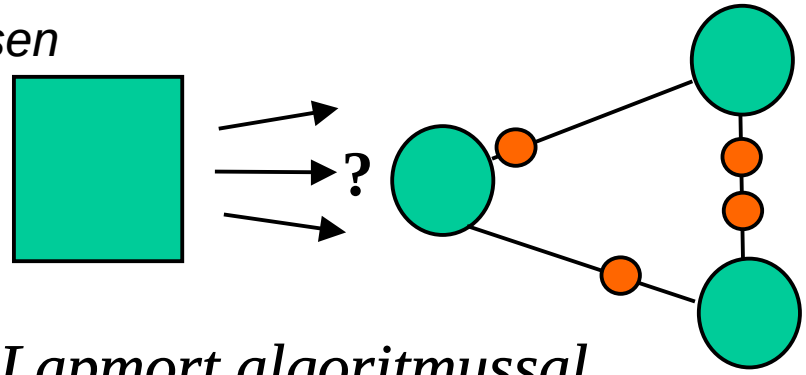




•Aszinkron checkpointoláshoz szignálok használtak
•Szignál kezelők a folyamatok indulásakor állítódnak be
•Pvm rutinok atomivá tételéhez a szignálok hatástalanítva vannak a rutinok elejétől a végéig
•Az atomi rutinok nem blokkoltá tételéhez a blokkoló rutinok használata nem engedélyezett
•A blokkoló rutinokat ismétlő nem blokkoltá alakítjuk

Módosítások IV.

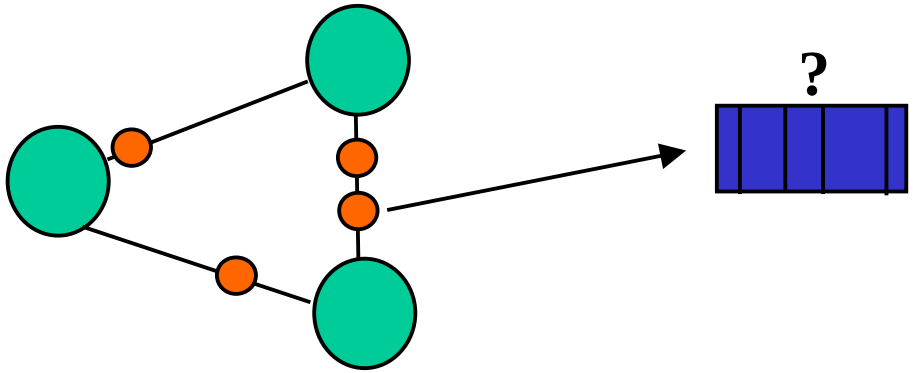
- *Úton lévő üzenetek kezelése*
  - *Mivel a pvm démonokat nem mentjük, a leváló folyamatok miatt üzenetek ragadhatnak a démonokban*
  - *A koordinációs folyamatnak tökéketesen ismernie kell a folyamatok számát*



•Úton levő üzenetek kezelése Chandy-Lapmort algoritmussal, üzenetek olvasása és tárolása, amíg üzenet-vége jel nem érkezik
•Ennek a protokollnak a futtatásához minden folyamatnak tökéletesen ismernie kell a szomszédjait ill. azok számát
•A folyamatok nyilvántartásához minden folyamat induláskor a teljes futási időn át tartó socket kapcsolatot nyit a koordinátorhoz
•Így akár az abortálást is detektálhatja a koordinátor

Módosítások V.

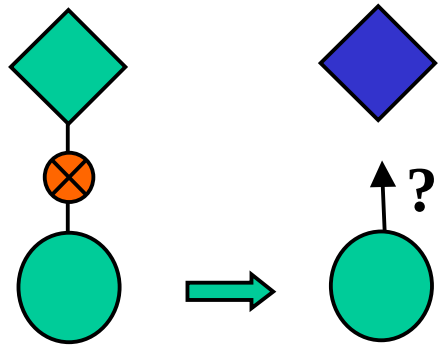
- *Úton levő üzenetek elmentése és el nem küldött bufferek*
  - *Úton levő üzenet elmentéséhez a megfelelő módon kell kicsomagolni azt*
  - *A kicsomagoláshoz ismerni kell az üzenet felépítését*
  - *El nem küldött üzenetbufferek kicsomagolásához is annak felépítése kell*



- *Üzenet formátum kinyeréséhez, módosított kódolást használunk*
- *Minden üzenetelem eltárolásakor a típus és méret is tárolva*
- *Az üzenet formátum ily módon részévé válik magának az üzenetnek*
- *“pack(5,int)” **è** “pack(1,(5 db int)); pack(5,int);”*

Módosítások VI.

- *(Újra)kapcsolódás a pvm démonhoz*
  - *Migráció után az új démonoknak új kapcsolati végpontja van*
  - *A végpontot fel kell deríteni, mert a visszaállított folyamat a régit tárolja*

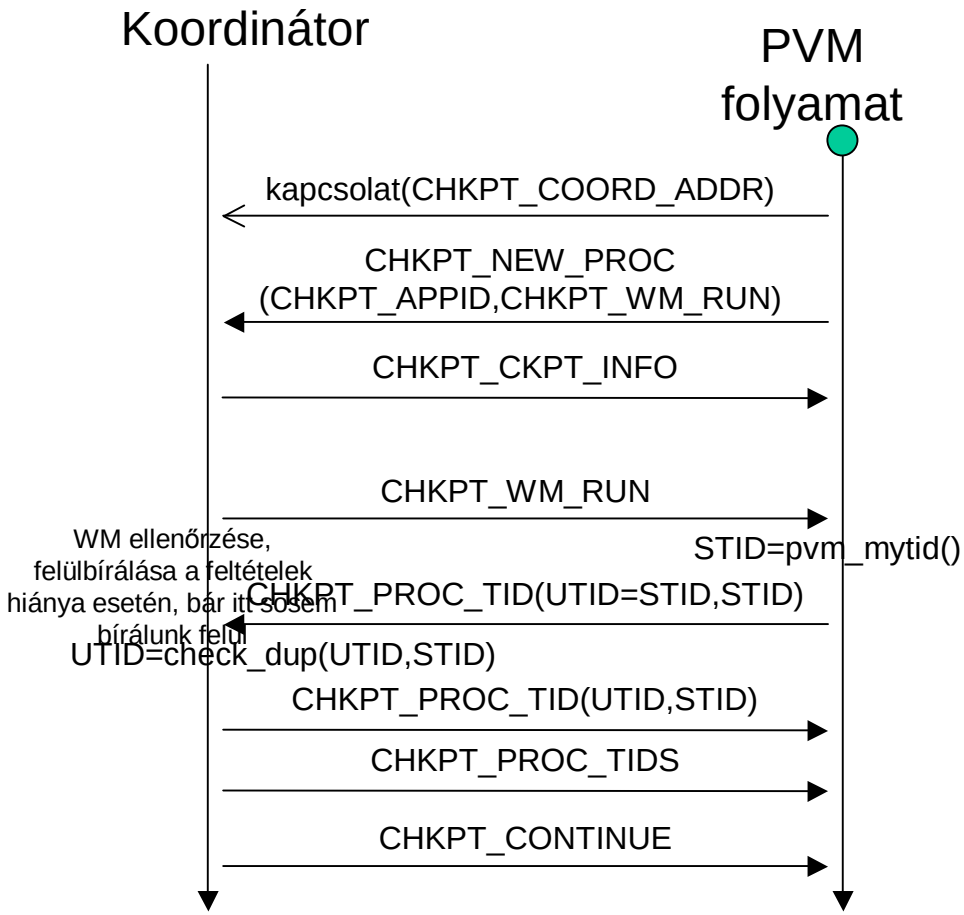


•A kapcsolat felépítő rutin (pvm_mytid) elé van beszúrva egy végpont felderítő algoritmus
•A végpont a visszaállítás előtti pillanatban eltárolható
•A végpont meghatározható a pvmd file tartalmából
•A végpont meghatározható a szülő folyamat nyitott végpontjainak szkennelésével is

Fontosabb protokollok a checkpoint/restart során

- Normál indítás
  - első folyamat: Feléledési protokoll 1.
  - gyermek folyamat: Feléledési protokoll 2.
- Visszaállítás
  - első folyamat: Feléledési protokoll 3.
  - gyermek folyamat: Feléledési protokoll 4.
- Lementési protokoll

Feléledési protokoll 1.
(első folyamat, normál indítás)



Környezeti változók:

CHKPT_COORD=<string>

default: checkpoint inaktiválása

CHKPT_APPID=<string>

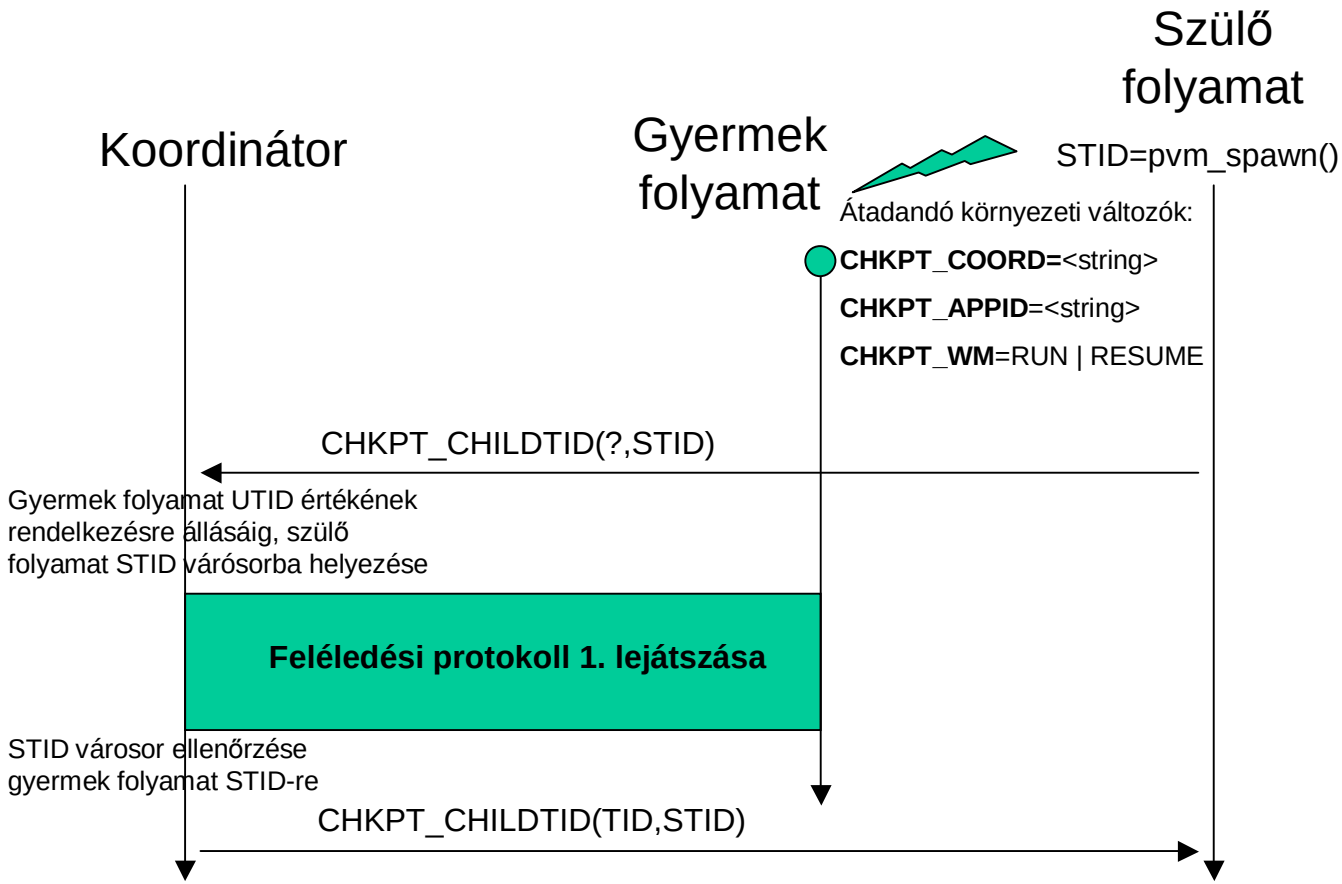
default: generálva

CHKPT_WM=RUN | RESUME

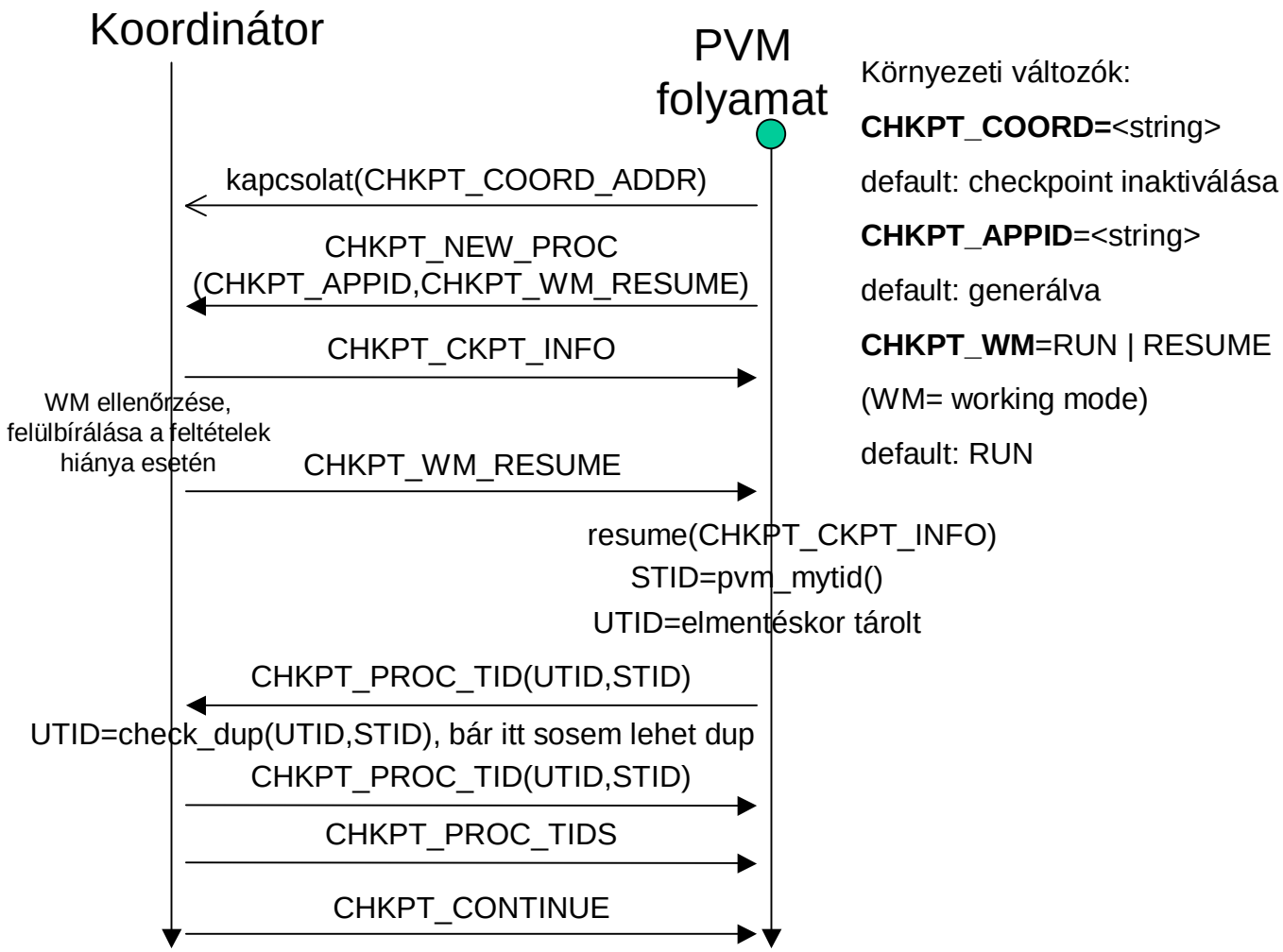
(WM= working mode)

default: RUN

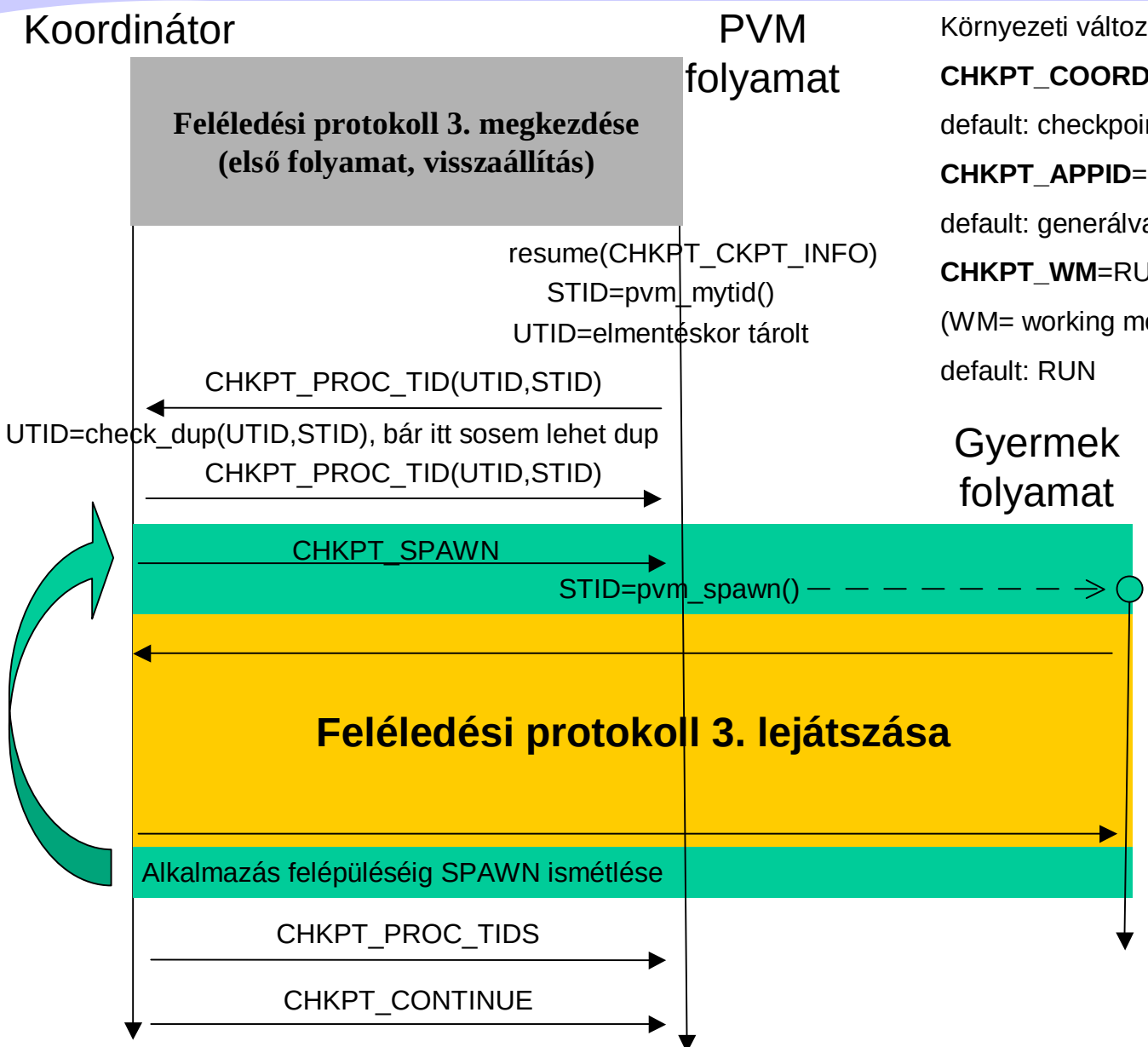
Feléledési protokoll 2.
(gyermek folyamat, normál indítás)



Feléledési protokoll 3.
(első folyamat, visszaállítás)



Feléledési protokoll 4. (gyermek folyamat, visszaállítás)



Környezeti változók:

CHKPT_COORD=<string>

default: checkpoint inaktiválása

CHKPT_APPID=<string>

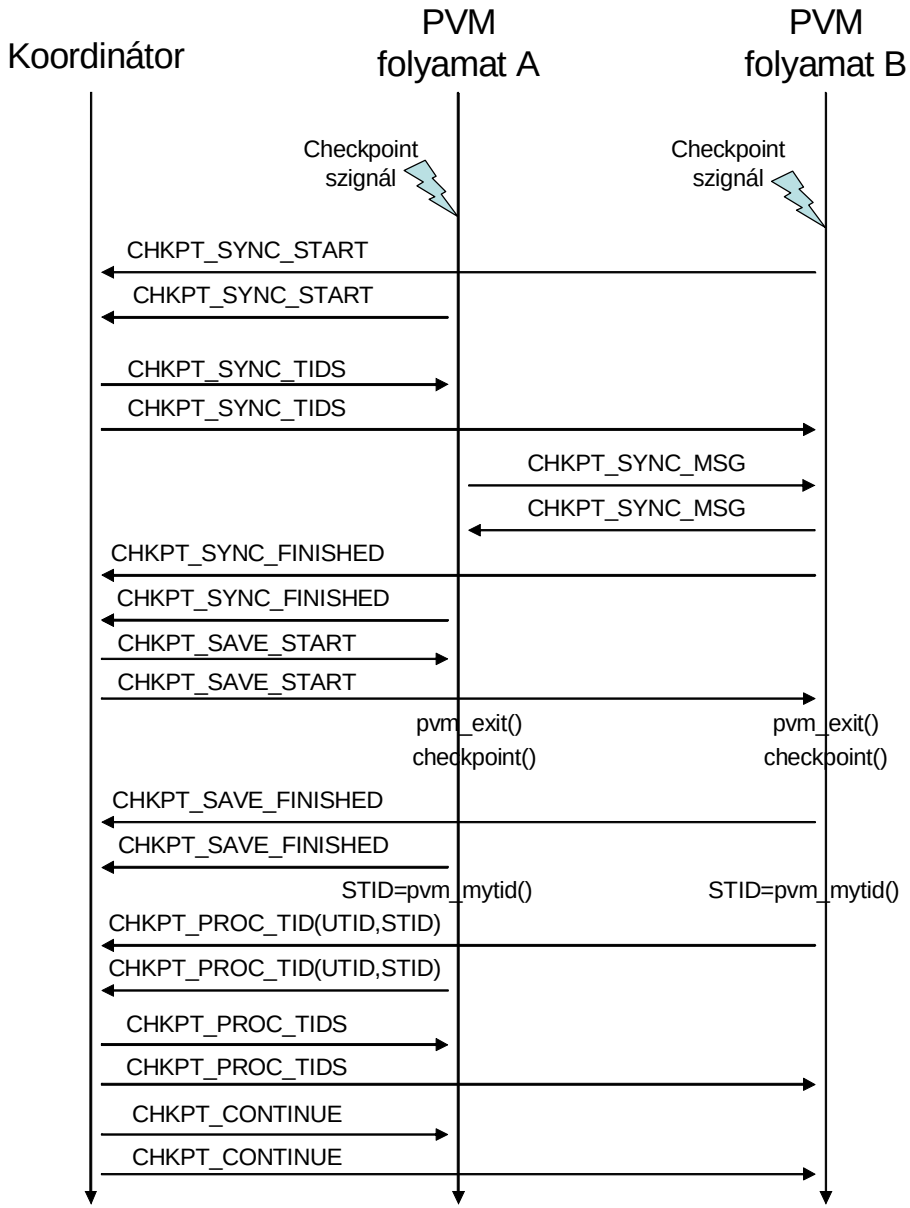
default: generálva

CHKPT_WM=RUN | RESUME

(WM= working mode)

default: RUN

Lementési protokoll



Összegzés

- a pvm checkpointolhatóságát a rutinok virtualizálásával és egy külső karmester segítségével oldottuk meg
- a megvalósítás nem igényli a pvm démonok módosítását
- a protokollok biztosítják az alkalmazás lementését és visszaállítását
- a pvm alkalmazások migrálhatók klaszteren belül és azok között
- az állapot leíró fájlok, checkpoint fájlok reprezentáljál az alkalmazás teljes állapotterét
- a bemutatott megoldás jelenleg fejlesztés alatt áll, hamarosan elkészül
- az elkészült checkpointoló a magyar KlaszterGrid-en lesz telepítve

Köszönöm a figyelmüket!

Kovács József

smith@sztaki.hu

http://www.lpds.sztaki.hu