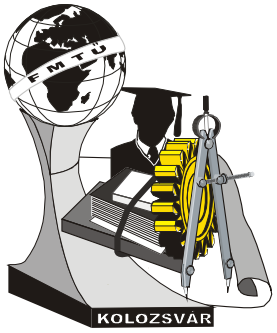
DOI: 10.36243/fmtu-1999.34




FIATAL MŰSZAKIAK TUDOMÁNYOS ÜLÉSSZAKA

Kolozsvár, 1999. március 19-20.

# WINPAR Párhuzamos programozási környezet Online kézikönyv és oktatási anyag

**Fülep Dávid**

**Abstract**
Within the framework of the WINPAR project the WINPAR integrated software development environment for parallel programming was elaborated. To make the WINPAR environment easy to use and easy to learn, a web site for development tools and environment was created. It provides information on the WINPAR project, on project partners and on on-line manuals and on-line tutorials of the WINPAR environment. The on-line manual is "slide-show" that presents services and usage of each tool showing how to use them step-by-step. The composition of the tutorial is similar to the on-line manual of tools, but in the on-line tutorial emphasise is placed on solving the problem not on how to use the tools. A mathematical problem was selected which can show the way of developing a parallel application. The Poisson problem is a simple partial differential equation that is at the core of many applications.

## 1. Bevezetés

Ahogy a számítógép-hálózatok a mindennapok számítástechnikai infrastruktúrájának részévé válnak, az osztott és párhuzamos számítógéprendszerek egyre több felhasználó számára válnak elérhetővé. A személyi számítógépek rohamosan növekvő teljesítménye lehetővé teszi egy osztott/párhuzamos programfejlesztő környezet PC-platformon történő kialakítását.

A WINPAR (WINdows based PARallel computing) projekt egy olyan integrált osztott és párhuzamos fejlesztőkörnyezet kifejlesztését tűzte ki célul, ami egy új típusú „párhuzamos" számítógépre épül: hálózatba kötött személyi számítógépekre. Operációs rendszerként a Windows NT-t választották.

Az osztott vagy párhuzamos rendszerben a kommunikáció legegyszerűbb módja az ún. üzenettovábbításos rendszer (message passing). Ez egy könnyen érthető és egyszerűen implementálható interface. A legfontosabb implementáció a PVM és az MPI. Mindkettő egy hardware-független, hatékony, rugalmas szabványt jelent. A PVM egyetlen virtuális gépben kezeli a különböző számítógépeket, az MPI egy szabványos interface-t biztosít az üzeneteken keresztül kommunikáló processzek számára.

Bár az üzenettovábbítás ezen módja könnyen áttekinthető, a párhuzamos programok fejlesztése a kommunikáció és szinkronizáció megszervezése miatt mégis sokkal bonyolultabb, mint a

szekvenciális programoké. Azok a programok, amik egyetlen processzoron hibátlanul futnak, váratlan módon viselkedhetnek többprocesszoros környezetben, többszöri futtatásra más és más eredményt adhatnak. További probléma párhuzamos programok nyomkövetése, debuggolása és tesztelése. Mindezek nem végezhetők el a hagyományos, szekvenciális programokhoz készült eszközökkel. Mindez alátámasztja egy osztott és párhuzamos programozást támogató fejlesztőkörnyezet szükségességét. A fejlesztés időszükségletének csökkentése mellett a hardware-költségek alacsonyan tartása a másik fontos cél, ez utóbbi miatt egy általánosan használható PC-rendszer jó megoldást jelent.

A jövő számítógépes szakembereinek a képzésében az osztott és párhuzamos programozás fontos témává vált. Az igényeknek megfelelő programozási környezet kulcsszerepet játszik az osztott és párhuzamos programozás oktatásában. A hallgatók alaposan tanulmányozhatják és megérthetik a program struktúráját és működését, amikor osztott debuggerrel elemzik annak futását. Az új programfejlesztési környezet oktatásba történő bevezetése azonban elképzelhetetlen megfelelő oktatási anyag nélkül. A kézikönyvek, tankönyvek, jól kiválasztott példák segítségével ismerhetik meg a hallgatók könnyedén a párhuzamos programozás világát.

## 2. A WINPAR környezet eszközei

A projekt fő célkitűzése a projekt partnerek által kifejlesztett fejlesztőeszközök egységes grafikus környezetbe integrálása, a WINPAR környezet online kézikönyvének és oktatási segédletének kidolgozása, és a párhuzamos programozásnak a Miskolci Egyetem oktatási rendszerében történő bevezetése volt. A WINPAR környezet három rétegre osztható: message passing réteg, ami az MPI és PVM implementációit jelenti; fejlesztőeszköz réteg, amiben minden egyes eszköz a fejlesztőkörnyezet egy részfeladatát látja el; és a programozási környezet, mint külön réteg, ami az első két réteget összefogva a komplett párhuzamos softwarefejlesztési környezetté teszi a rendszert.

### 2.1 Message Passing eszközök

- WPVM (Windows Parallel Virtual Machine). Ezen programcsomag segítségével egyetlen ún. virtuális gépként kezelhető a hálózat. A fejlesztő az alkalmazást együttműködő taszkokból készíti el, az egyes taszkok a PVM erőforrásokat pedig szabványos könyvtárból veszik.
- MPI (Windows-based Message Passing Interface). Az MPI újabb fejlesztés, egy minden igényt kielégítő függvénykönyvtárral segíti a processzek közötti (hálózati vagy gépen belüli) kommunikáció hatékony megvalósítását.

### 2.2. Fejlesztőeszközök

- TRAPPER. A WINPAR környezet egyik legalapvetőbb eszközével végezhető a hardware és software modell megtervezése, a processzek leképezése a processzorokra (mapping), a tényleges kód előállítása, majd a futtatás és tesztelés (monitoring, vizualizálás).
- AUGUR (AUtomatic model Generation with User Response), és MODARCH. E két eszköz segítségével végezhető a párhuzamos programok teljesítményének kiértékelése automatikus modellgenerálással, illetve szimulációval.
- DIWIDE. A WINPAR környezet hibakereső eszköze.

### 2.3. Programfejlesztő környezet

- WINPARK NAVIGATOR. Ez a speciális eszköz vezérli az eszközök együttműködését, minden egyszerűen indítható. A feladatokat szellemes módon egy térképen jeleníti meg, és irányítja a felhasználót a helyes programfejlesztési lépések kiválasztásában. A térképen az egyes épületek a fejlesztés különböző stádiumait jelentik, ami között egy kis autóval „közlekedhet" a fejlesztő. Az éppen nem kiválasztható munkafázisok (épületek) előtt „parkolni tilos" jelzés van.

## 3. WINPAR Web-szerver

A WINPAR fejlesztőkörnyezethez kapcsolódó anyagok tárolására egy web-szervert állítottunk fel. A projekt általános leírásán és a projekt partnerek bemutatásán kívül itt található az egyes eszközök online kézikönyve és a párhuzamos programozáshoz készített oktatási segédanyag. Mindkettő felépítésére jellemző, hogy a legapróbb részletekig kitér minden egyes fontos lépésre, legyen szó egy adott eszköz használatáról vagy a párhuzamos programozás elméleti kérdéseiről. Az eszközök működésének begyakorlása és a tananyag elsajátítása online tesztek kitöltésével ellenőrizhető. Ez a felhasználói visszacsatolás igen fontos a projekt szempontjából, az így szerzett tapasztalatok később felhasználhatók. A projekt elvárásainak megfelelően a web-szervert is egy Windows NT gépen alakítottuk ki. A UNIX világból ismert Apache szervert választottuk. A web-szerver címe: http://winpar.iit.uni-miskolc.hu

## 4. Online kézikönyv és oktatási anyag

Az online kézikönyv (manual) az egyes eszközök teljes körű bemutatását tartalmazza. Miközben a felhasználó a magyarázatokat olvashatja, mindvégig az eredeti eszköz felhasználói felületét látja maga előtt, és a továbblépés is azon keresztül történik. A gyakorlott felhasználók a már megismert részeket könnyedén átléphetik.

Az online oktatási segédlet (tutorial) lépésről lépésre vezeti végig a felhasználót az osztott és párhuzamos programozás folyamatán. Az alapvető különbség az eszközök kézikönyveivel szemben az, hogy amíg ott az eszközök kezelésére koncentráltunk, addig itt a megoldandó problémán van a hangsúly. A fejlesztés menetét természetesen konkrét példák segítségével lehet legkönnyebben megérteni. Egy olyan matematikai problémát választottunk, ami sok alkalmazás alapját képezi. Az ún. Poisson-probléma egy egyszerű parciális differenciálegyenlet. Ennek megoldása során lineáris egyenletrendszert kell megoldani, ami az interaktív Jacobi módszerrel könnyen párhuzamosítható.

A párhuzamos programfejlesztés főbb lépései a következők: A software és hardware struktúra megtervezése, majd a tényleges funkció leprogramozása. Ezután az alkalmazás processzeit az egyes fizikai processzorokhoz hozzá kell rendelni (mapping). Ekkor következhet a program lefordítása és futtatása. A kommunikációt végző könyvtárak, az egyes taszkok kódkeretei, és a mindehhez szükséges konfigurációs állományok generálását a rendszer automatikusan elvégzi a grafikus leírások alapján, a felhasználónak csak a program tényleges működését jelentő részét kell magának megírnia. Sikeres fordítás után következik a tesztelési fázis, ami a helyesség és a hatékonyság vizsgálatát jelenti. Az ezekhez szükséges monitorozó, vizualizáló, hibakereső, modellező és szimulációs eszközök rendelkezésre állnak.

A probléma megoldása közben, ha szükséges, bárhol visszatérhetünk az adott részfeladatot megvalósító eszköz kézikönyvének megfelelő részéhez, hogy megtanuljuk, hogyan kell majd a feladatot elvégezni a gyakorlatban. Az ismert, vagy kevésbé fontos részek a tanulás alatt kihagyhatók, az anyag a hallgatók és a fejlesztők igényeinek megfelelő szabad sorrendben dolgozhatók fel. Természetesen ajánlott útvonalak segítik a teljes rendszer következetes megismerését.

## 5. Irodalomjegyzék


- Henri Ball: Programming Distributed Systems /Silicon Press, 1990./
- Andrew Goscinski: Distributed Operating Systems - The Logical Design /Addison Wesley, 1992./
- George Coulouris, Jean Dollimore, Tim Kindberg: Distributed Systems /Addison Wesley, 1994./
- Andrew S. Tanenbaum: Distributed Operating Systems /Prentice-Hall, 1995./



Fülep Dávid /okleveles informatikus mérnök, Phd. hallgató/
Miskolci Egyetem, Általános Informatikai Tanszék /3515 Miskolc-Egyetemváros/
Email: fulep@iit.uni-miskolc.hu
Tel: +36 - 46 - 565111 / 19-52