Synchronizing Inforum Models

Frank Hohmann

GWS mbH Osnabrück, Germany

12th INFORUM World Conference, Italy 2004

Overview

(1) Project Description
(2) Approaches for Linking Models
(3) Inter-Process Communication (IPC) functions on Win32 platforms
(4) Passing Data between Models
(5) Control Flow in Synchronized Models
(6) Conclusion

(1) Project Description

- Project financed by IAB (Institut für Arbeitsmarkt und Berufsforschung, engl. Institute for Employment Research)
- Goal: Linkage of two models
  - GINFORS (**G**lobal **In**terindustry **For**ecasting **S**ystem)
    - Part of EU project MOSUS (Modeling Opportunities and Limits For Restructuring Europe towards Sustainability)
    - Built for forecasting energy use, CO2 emissions, material consumption and land use
    - Uses official data sources only (like OECD)
    - Contains 53 country models (20-30 with IO)
  - INFORGE (**In**terindustry **For**ecasting **Ge**rmany)

(2) Approaches for Linking Models (Page 1 of 2)

1. Solving models in batch mode, linkage through databases
   - \+ Models can be maintained / improved independently
   - \- Models do not solve simultaneously
   - \- Solving is slow
2. Merging models‘ sources (& databases)
   - \- Models can NOT be maintained / improved independently
   - \+ Models solve simultaneously iteration by interation
   - \+ Solving is fast

(2) Approaches for Linking Models (Page 2 of 2)

1. Synchronizing models through IPC (Inter-Process Communication) functions
   - IPC functions are available on multi-tasking operating systems (like Win2K/XP, Linux)
   - Used for synchronizing processes (e.g. word processor and printer spooler)
   - Approach combines advantages of 1. and 2., avoids disadvantages, thus:
     + Models can be maintained / improved independently
     + Models solve simultaneously on iteration-by-interation basis
     + Solving is fast

(3) Inter-Process Communication (IPC) functions on Win32 platforms

- CreateEvent
  - Creates an event data type for sending signals between processes, identified by a string
- SetEvent / WaitForSingleObject
  - Sends / waits for a signal
- CreateFileMapping / OpenFileMapping
  - Creates / opens a shared memory area, identified by a string
- MapViewOfFile
  - Obtains a pointer to the shared memory area

(4) Passing Data between Models (Page 1 of 2)

- Interdyme data structures (Tseries, Vector, Matrix) are not known at operating system level
  - Define a struct containing a list of variables to be shared

```
typedef struct _SharedData
{
  float TSVar;                  // Tseries variable
  float VecVar[n];              // Vector variable
  float MatVar[n][m];           // Matrix variable
  …
  // Flags indicating whether a model converged or not
  bool isGinforsReady, isInforgeReady;
} SharedData;
```

(4) Passing Data between Models (Page 2 of 2)

- Type-cast pointer returned by MapViewOfFile to point to previously defined data structure

```
SharedData *psd = (SharedData*) MapViewOfFile(…);
```

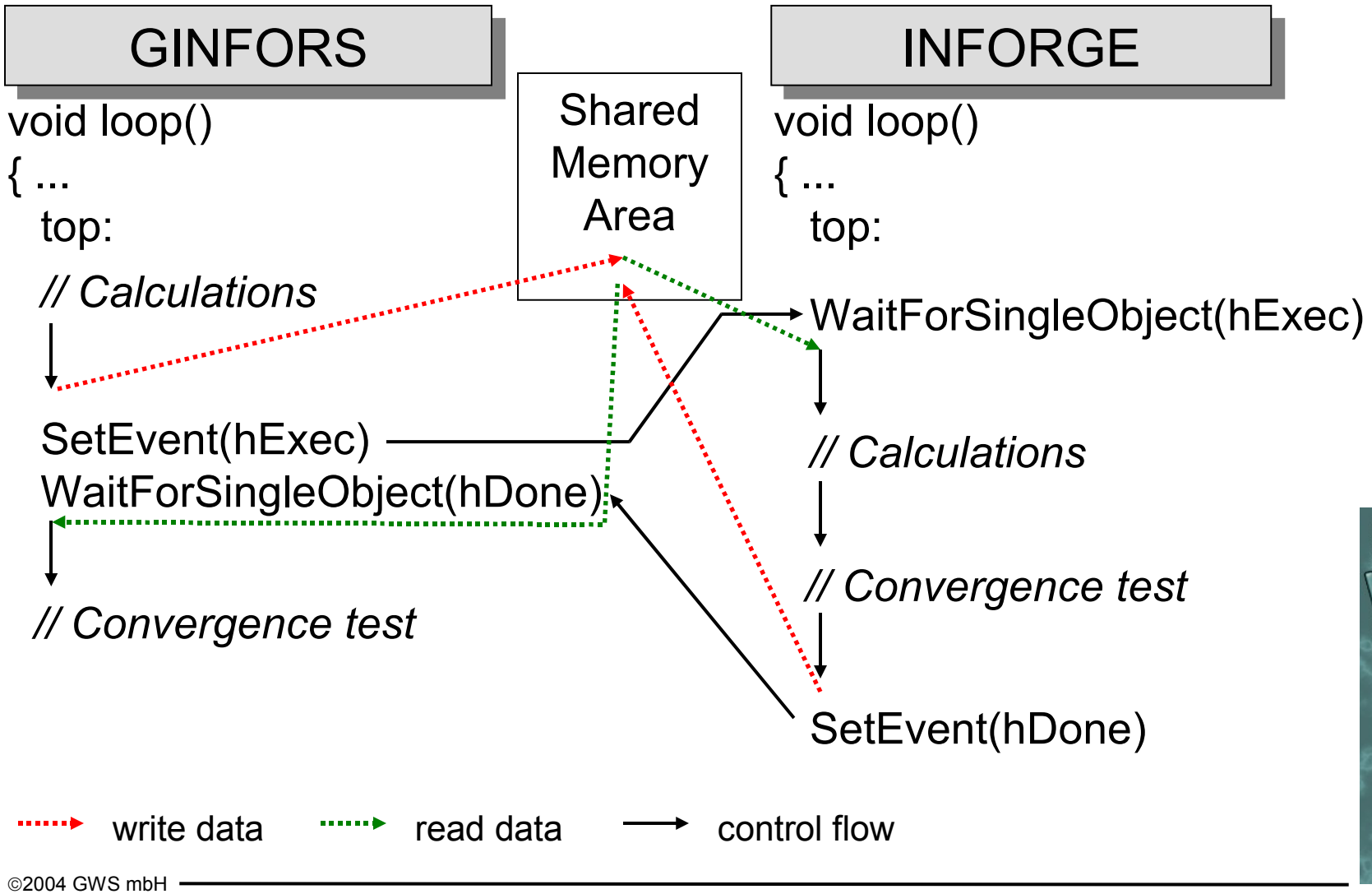
- Shared variables can now be accessed as follows:
  - Reading data (LHS: Interdyme, RHS: Shared Memory)

```
TSVar          = psd->TSVar;
VecVar[n]      = psd->VecVar[n];
MatVar[n][m]   = psd->VecVar[n][m];
```

  - Writing data (LHS: Shared Memory, RHS: Interdyme)

```
psd->TSVar          = TSVar;
psd->VecVar[n]      = VecVar[n];
psd->MatVar[n][m]   = VecVar[n][m];
```

(5) Control Flow in Synchronized Models



(6) Conclusion

- Models can by synchronized with minimal effort
  - ~ 10 lines of code have to be added to each model
- Models can be maintained / improved independently
  - Synchronization can be deactivated by using #ifdef‘s
- Synchronization can be used for lots of applications

Country model	synced to	Global model
Regional model	synced to	Country model
Country model	synced to	Country model