IMPLEMENTATION OF PROCESS NETWORKS IN JAVA

Arnab Basu

Vijay Kishen

EE382C-9

Embedded Software Systems

Instructor: Dr Brian Evans

University of Texas at Austin

Project Goals

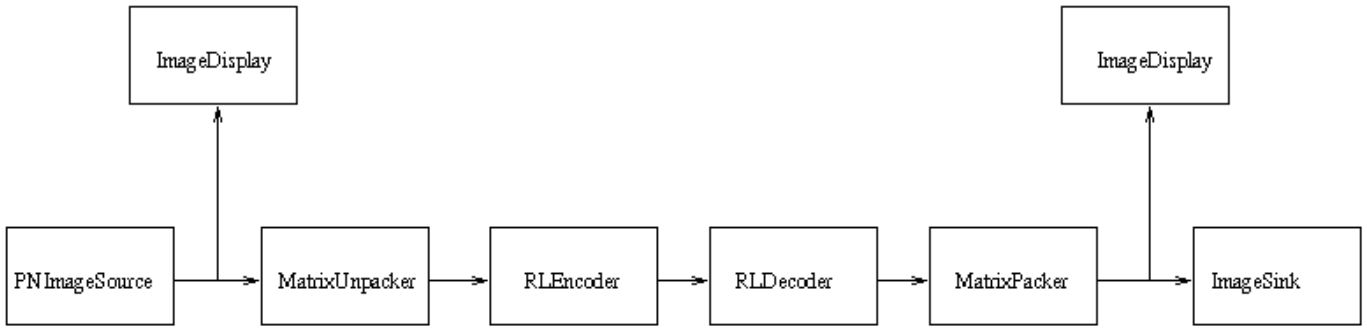
- Design and Implementation of a PN Framework
- Deadlock Detection and Resolution
- Test Framework with Actors
- Performance Evaluation

Approach and Implementation

- **Queues**
  - **Enqueue, Dequeue**
  - **increaseSize**
- **Event Based Programming**
  - **EventObject - PNBlockedEvent**
  - **Listeners - PNBlockedEventListener**
- **Actor**
  - **listens for blocked events**
  - **deadlock/write block resolution**

Test Framework with Actors

- **Computational PN Actors - Greg Allen.**
- **The PN Demo from Ptolemy II**



Deadlock Resolution and Performance Evaluation

Queue Size	10	100	1000
Parks			
Time(secs)	**319.33**	**326.25**	**368.17**
Memory	**4718**	**5378**	**9541**
Kahn			
Time(secs)	**115.84**	**150.6**	**183.32**
Memory	**5015**	**5879**	**15681**

Set Up: IBM PC AMDK6 400MHz, 128 Mbytes RAM, OS: WIN98

Conclusions

- Park's PN implementation assures bounded execution but at the cost of execution speed
- Selection of initial capacity of queues is of critical importance can be improved with "tweaking"
- Trade off execution time and memory

References

- **Ptolemy II design Documentation**

http://ptolemy.eecs.berkeley.edu/publications/papers/99/HMAD

- **Greg Allen's Computational Process Network Source Code**

  http://www.ece.utexas.edu/~allen/CPNSourceCode/index.html