

# Performance and Scalability of EJB Application

Emmanuel Cecchet  
Julie Marguerite  
Willy Zwaenepoel

Presented by Chen Fu

# Unfolding J2EE

- Servlets

} TomCat

- EJB

- Session Beans

- Stateless
- Stateful

- Entity Beans

- Bean Managed Persistence ==> Program managed persistence
- Container Managed Persistence ==> Framework managed persistence

Jboss  
JonAS

Web  
Application  
Server

(WebSpher

# Servlets



<http://cgi.ebay.com/ws/eBayISAPI.dll?ViewItem&category=2990&item=436654343>

# Servlets

<http://cgi.ebay.com/ws/eBayISAPI.dll?ViewItem&category=2990&item=436654343>

The request gets to TomCat

# Servlets

<http://cgi.ebay.com/ws/eBayISAPI.dll?ViewItem&category=2990&item=436654343>

Used by TomCat to find the right Servlet Class

# Servlets

<http://cgi.ebay.com/ws/eBayISAPI.dll?ViewItem&category=2990&item=436654343>

Parameters Passed to the Servlet Cl

# Servlets

<http://cgi.ebay.com/ws/eBayISAPI.dll?ViewItem&category=2990&item=436654343>

```
public class SomeServlet extends HttpServlet{
    public void init(ServletConfig config) throws ServletException{
        ...
    }

    public void destroy() {
        ...
    }

    protected void doGet(HttpServletRequest req,
                          HttpServletResponse resp)
        throws ServletException,
        java.io.IOException{
        ...
    }
    ...
}
```

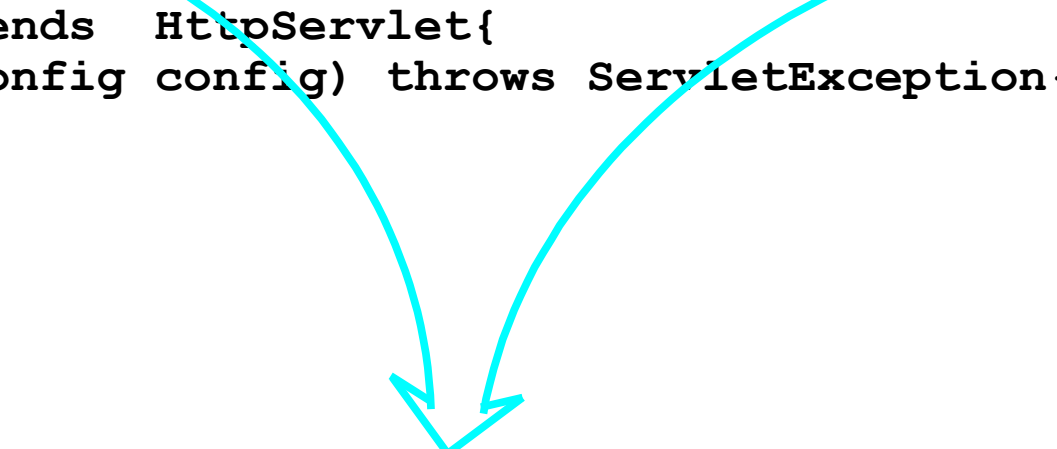
# Servlets

<http://cgi.ebay.com/ws/eBayISAPI.dll?ViewItem&category=2990&item=436654343>

```
public class SomeServlet extends HttpServlet{
    public void init(ServletConfig config) throws ServletException{
        ...
    }

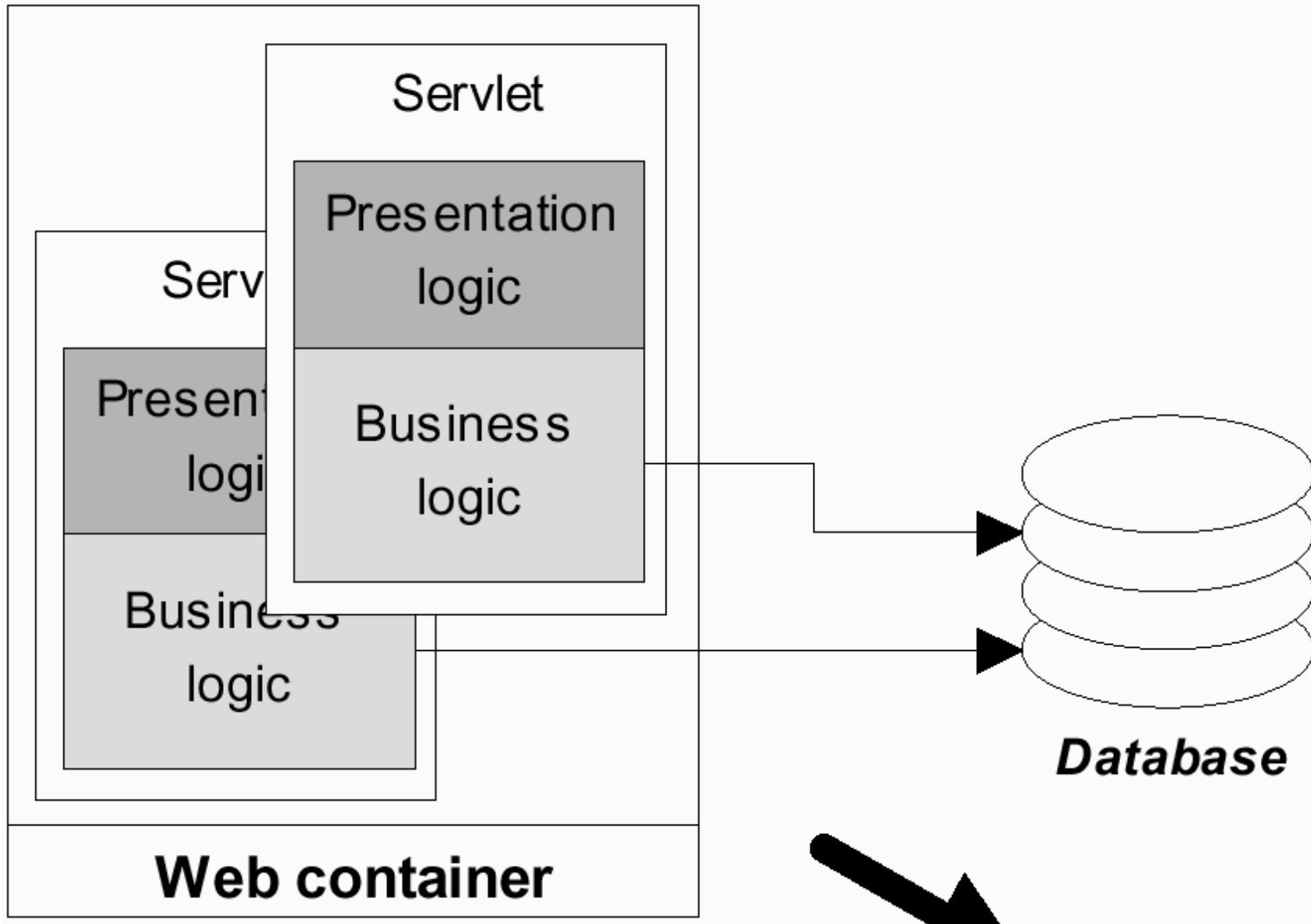
    public void destroy() {
        ...
    }

    protected void doGet(HttpServletRequest req,
                          HttpServletResponse resp)
        throws ServletException,
        java.io.IOException{
        ...
    }
    ...
}
```





# Servlets



# EJB

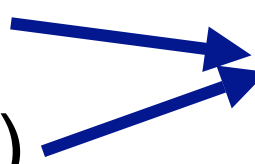


---

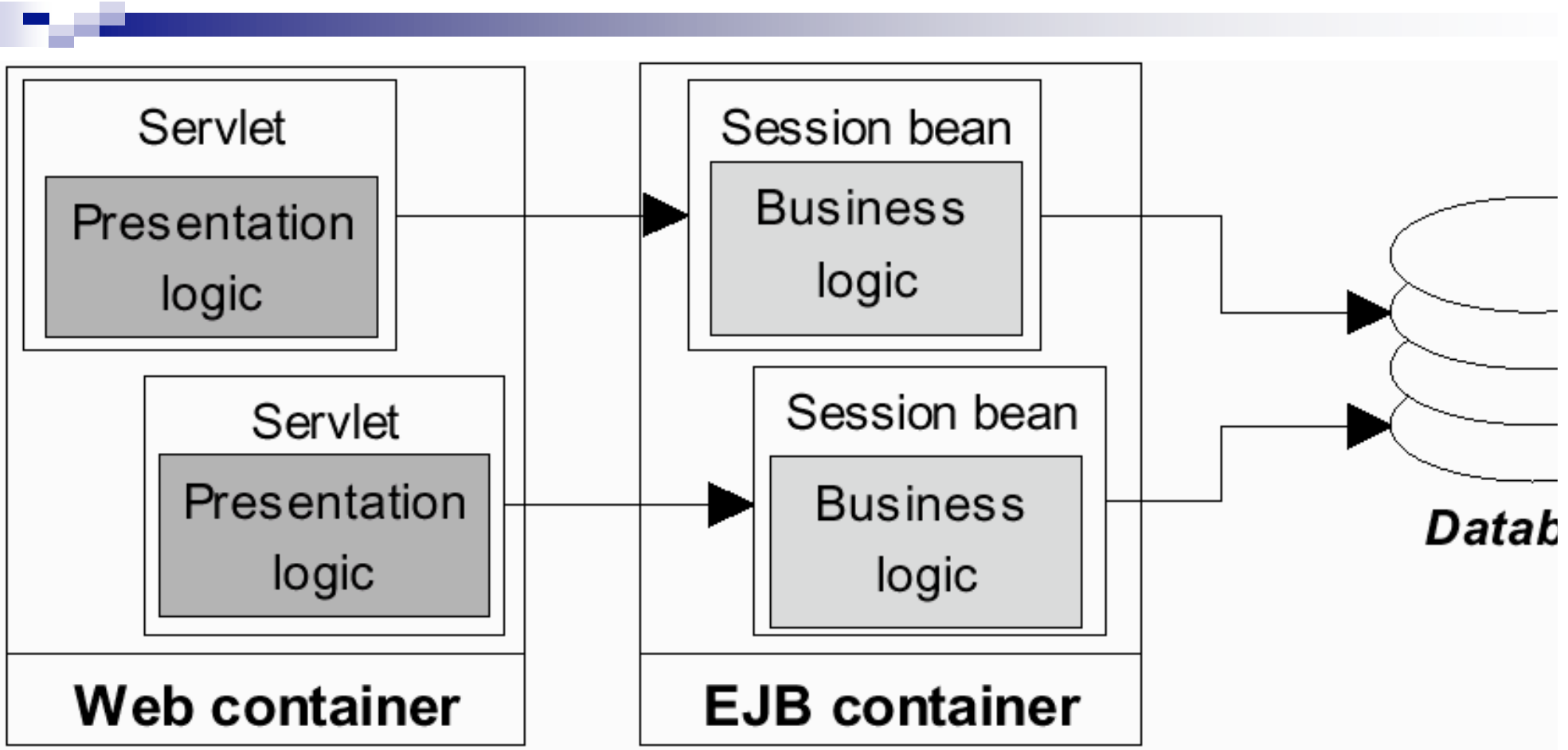
- Implementation of RMI server with some life cycle management
  - A bean class can post some interface for remote client (usually servlets) to call via RMI.
  - A bean class needs to implement some methods to be informed of some events:
    - Session Beans
    - Entity Beans

# Session Beans

## ■ Session Beans

- void ejbActivate() 
- void ejbPassivate()
- void ejbRemove()
- void setSessionContext(SessionContext ctx)

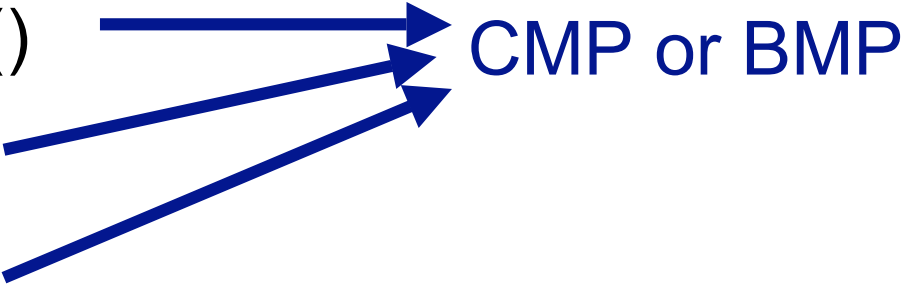
# Session Beans



“Use connection pooling transaction management provided by the EJB server”

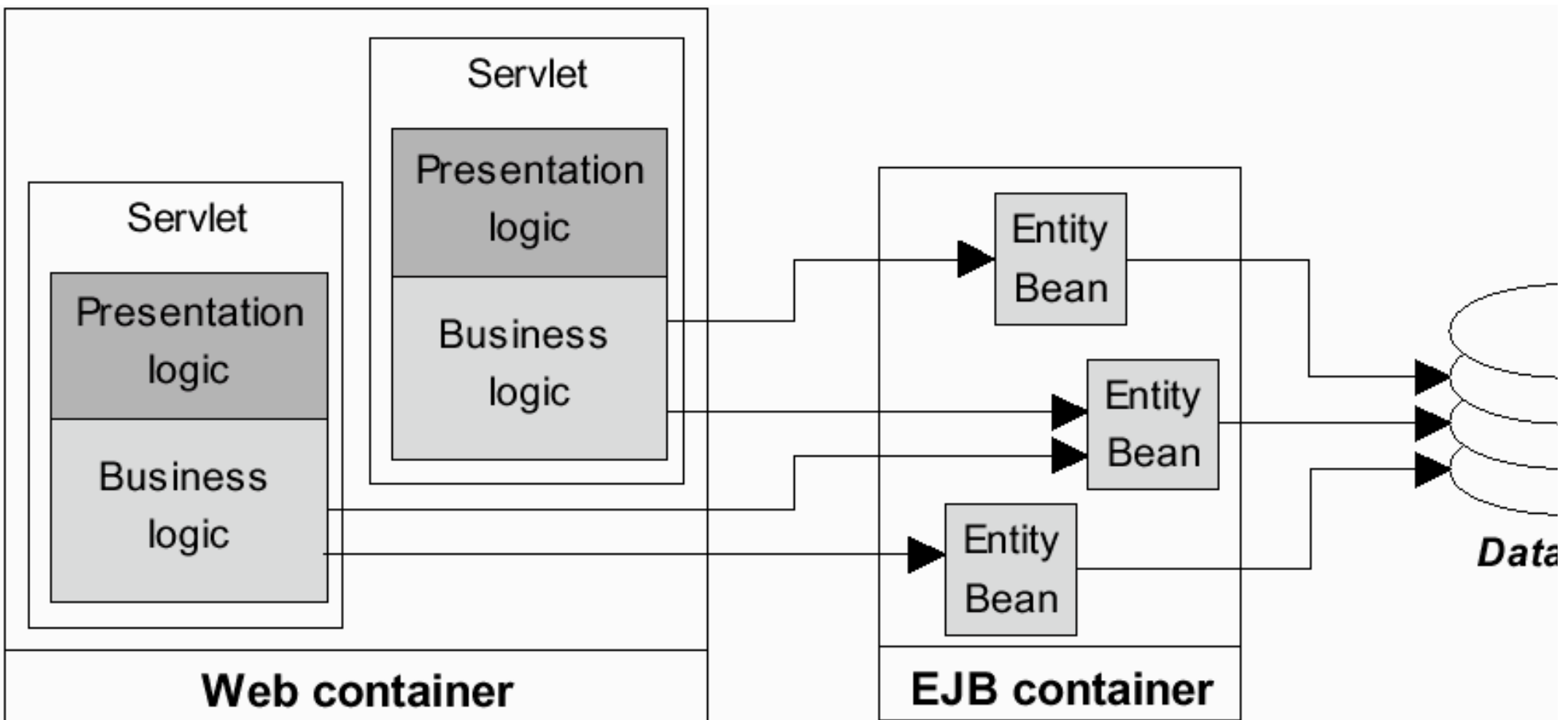
# Entity Beans

## ■ Entity Beans

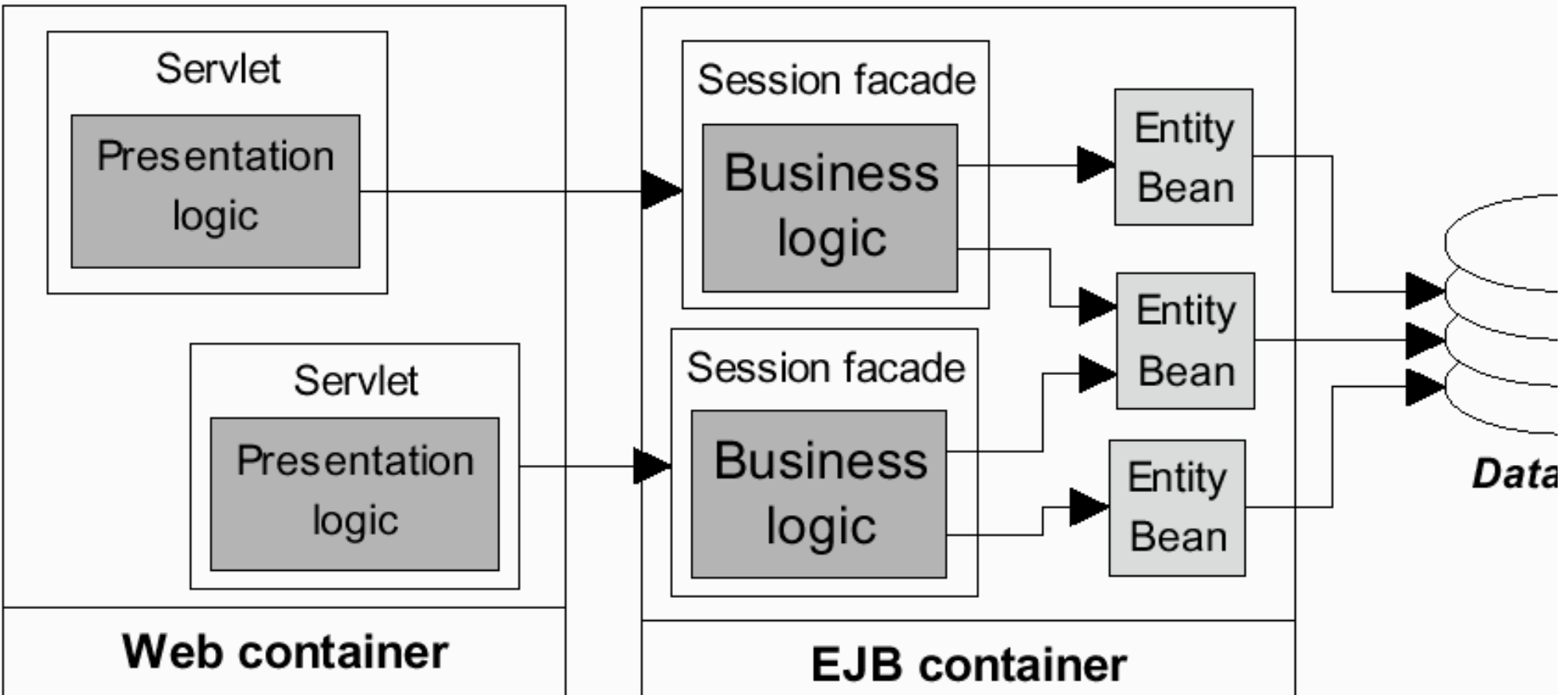
- void [ejbActivate\(\)](#)
  - void [ejbPassivate\(\)](#)
  - void [ejbRemove\(\)](#)
  - void [ejbLoad\(\)](#)
  - void [ejbStore\(\)](#)
  - void [setEntityContext\(EntityContext ctx\)](#)
  - void [unsetEntityContext\(\)](#)
- 
- The diagram shows three blue arrows pointing from the right side of the `ejbRemove()`, `ejbLoad()`, and `ejbStore()` method signatures to the text "CMP or BMP".

- Each entity bean usually associate with one row from some table in the database

# Entity Beans



# J2EE Framework



# Configuration Parameters

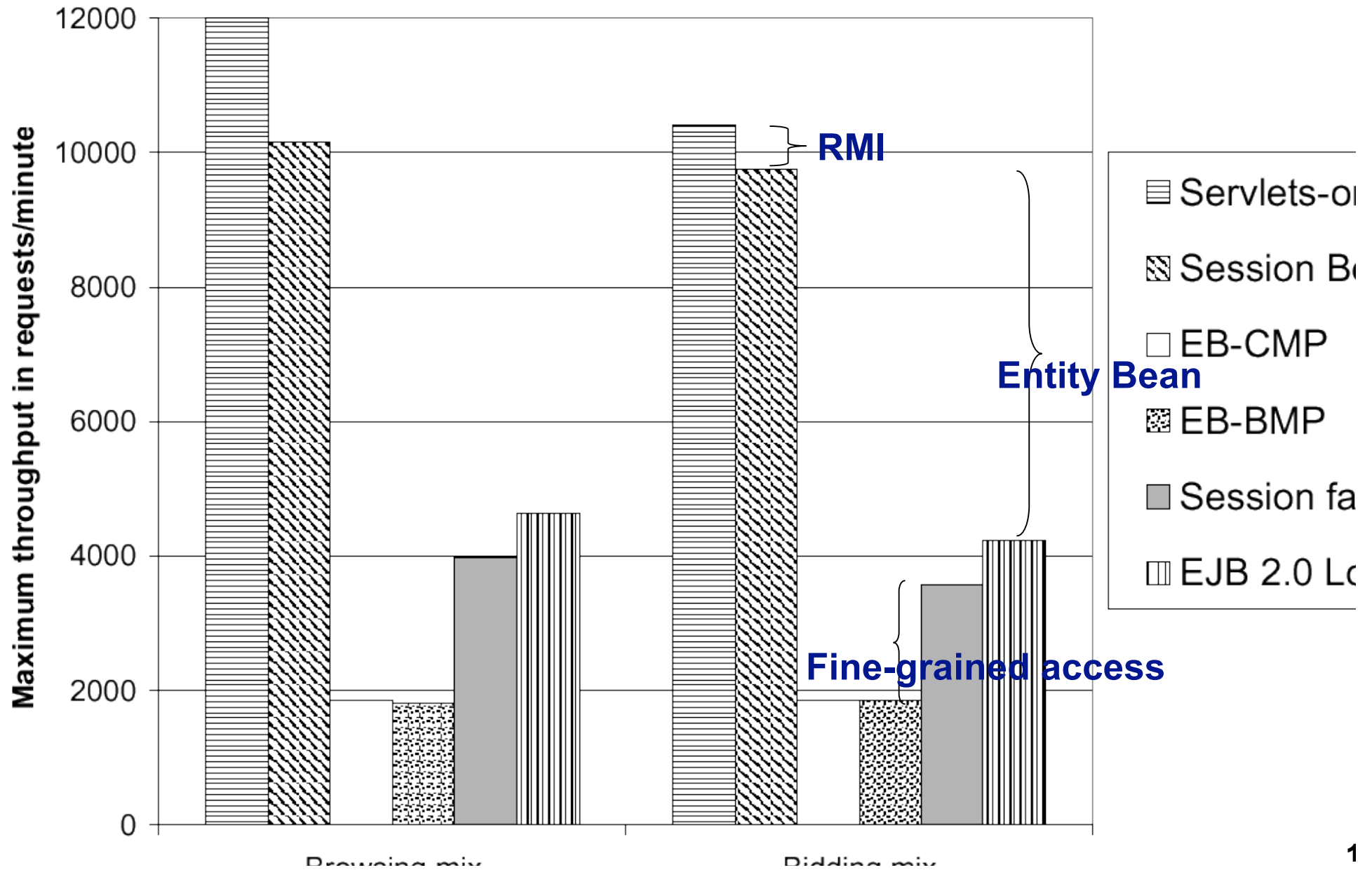
- Implementations of DynaServer
  - Servlets Only
  - Servlets + Session Beans
  - Servlets + Entity Beans
    - BMP
    - CMP
  - Servlets + Session Beans + Entity Beans (CMP)
    - With Local Interface
    - Without Local Interface
- Implementation of Container
  - Proxies
    - Pre-compiled
    - Reflection
  - Communication
    - General RMI
    - Jeremie  
(communication optimization)



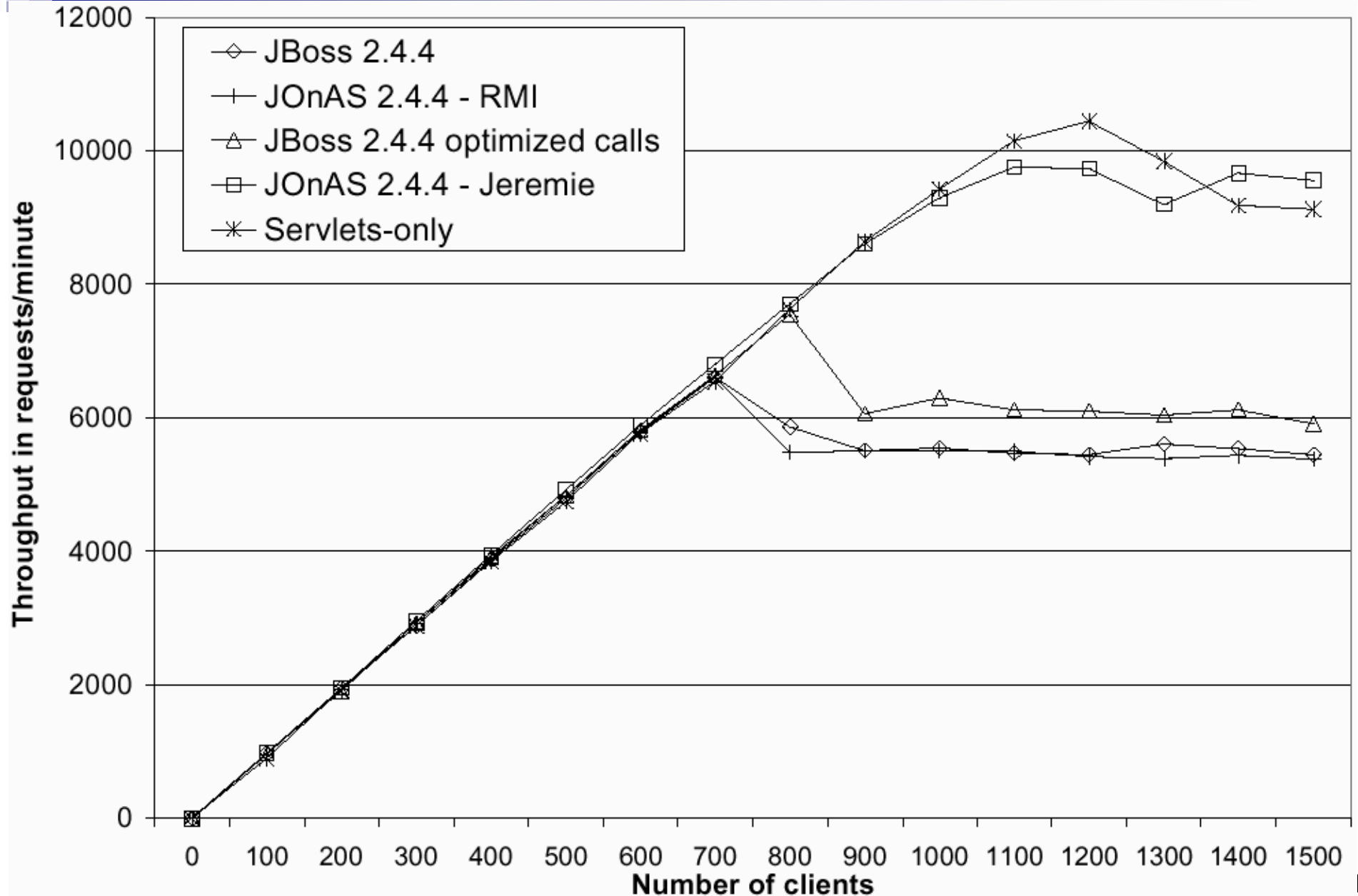
# Code Base

	Servlets		Beans		Total	
	Classes	Lines of code	Classes	Lines of code	Classes	Lines of code
Servlets-only	25	4590	-	-	25	4590
Session beans	22	2730	51	5270	76	8000
EB CMP	23	3980	40	6780	63	10760
EB BMP	23	3980	40	9850	63	13830
Session façade	22	2660	85	10780	107	13440
EJB 2.0 local	22	2725	91	11070	113	13795

# Max Throughput for

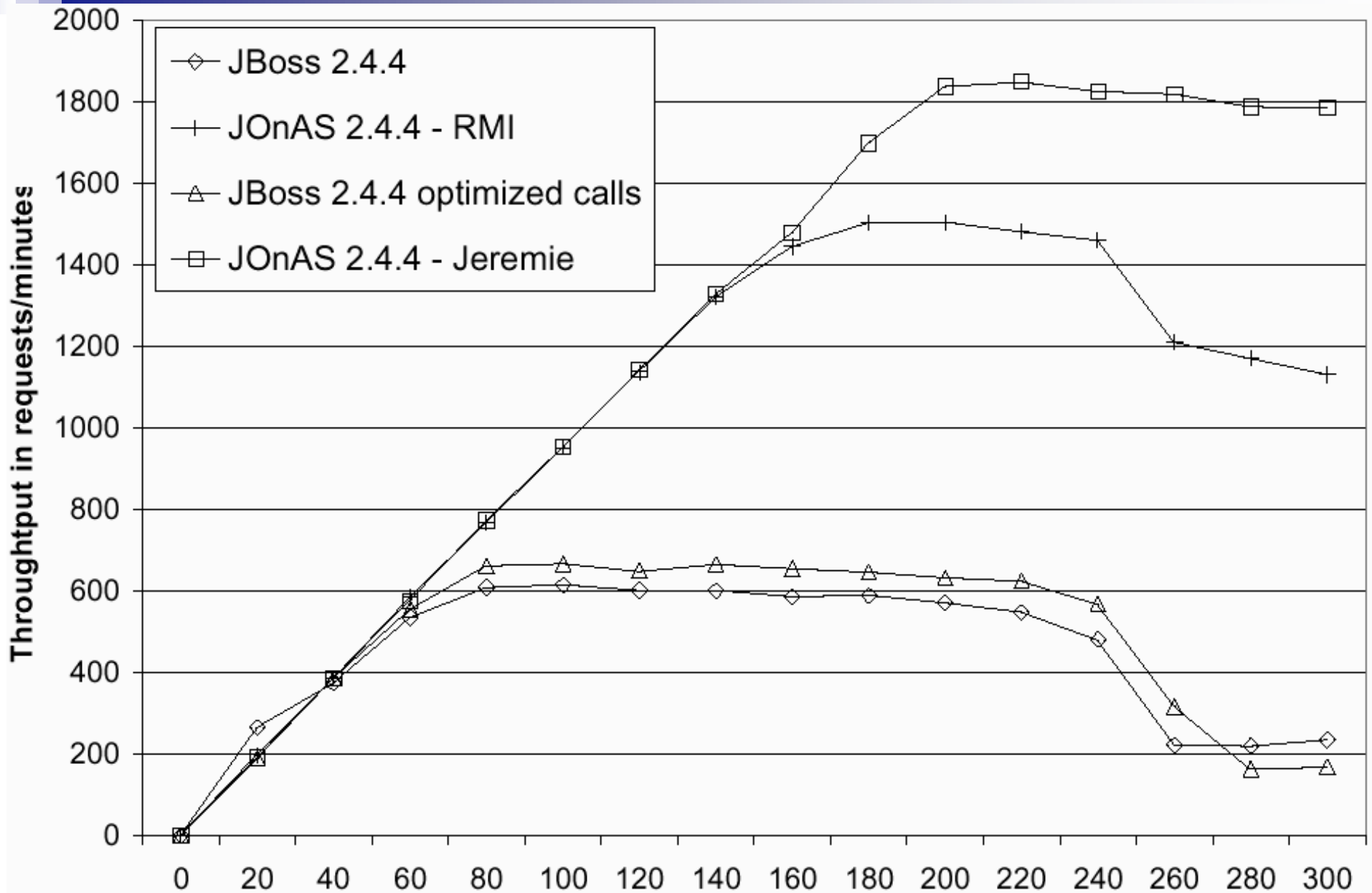


# Session Beans -- scalability



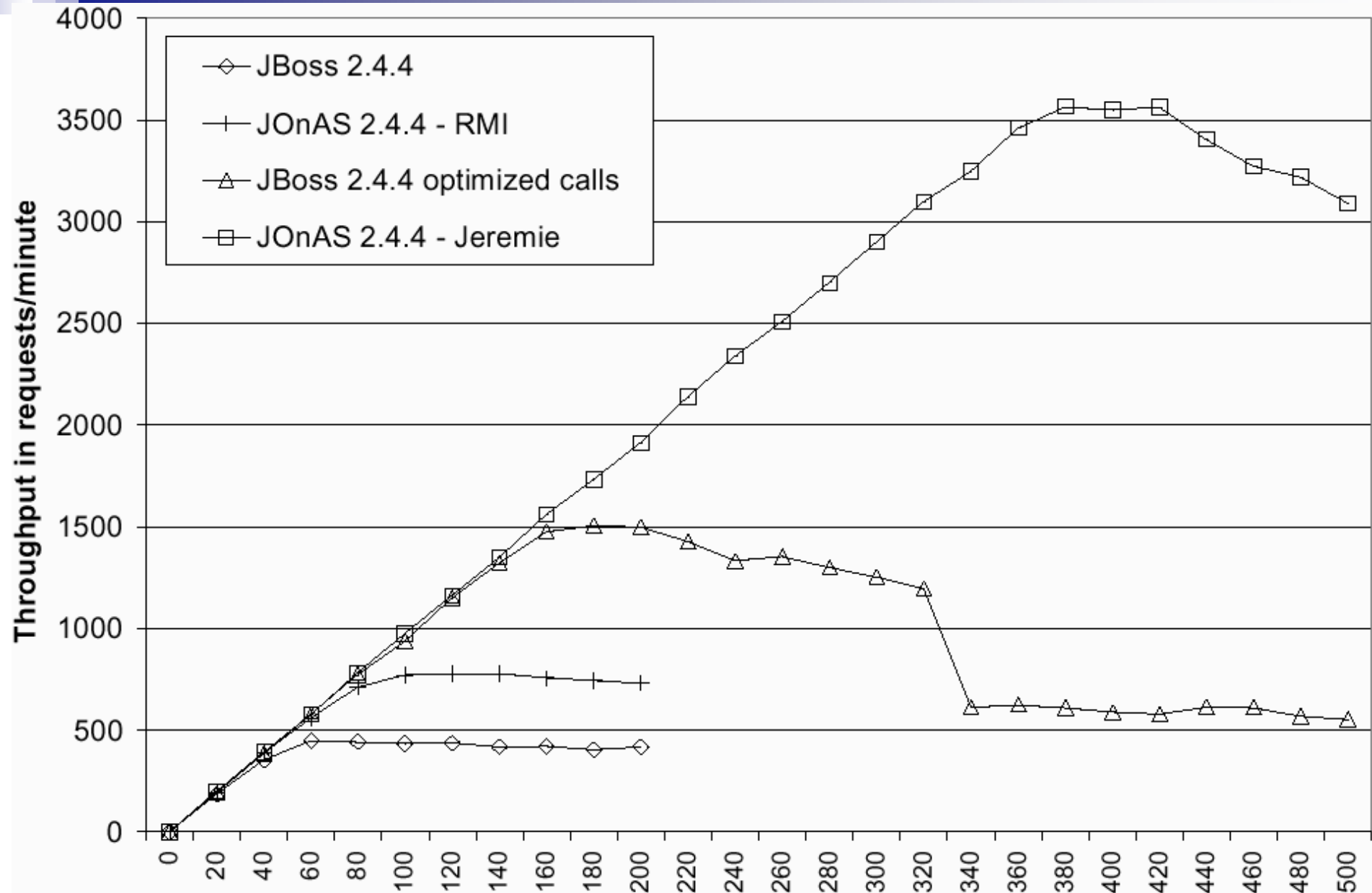


# Entity Beans – scalability





# Session Façade -- scalability







# Other Factors

---

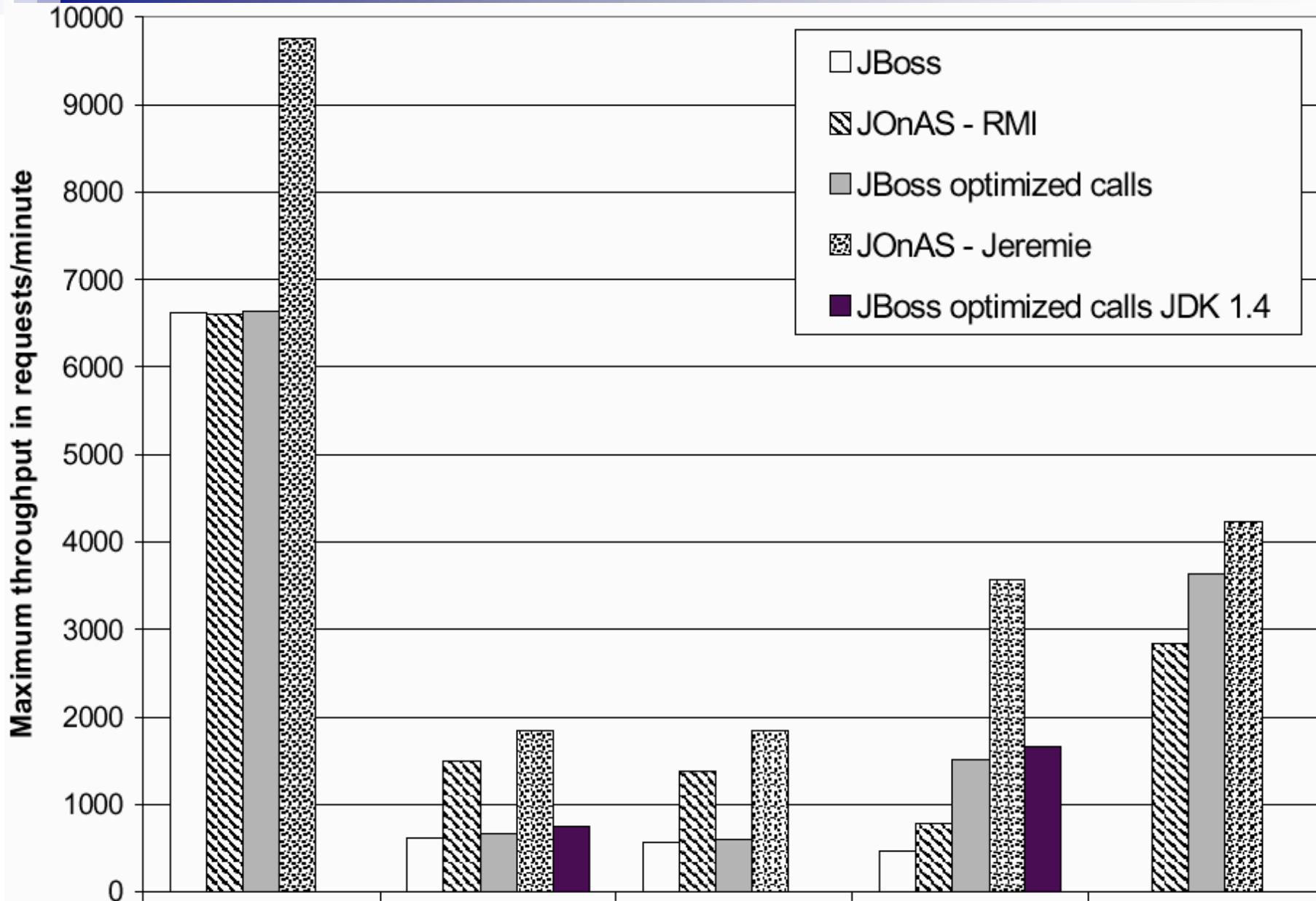
- Local Interface

- Get performance gain in all configuration, (optimized communication or not) Why?

- JDK 1.4

- Reflection speed up is moderate
- Other part actually slow down
- No significant improvement

# Summary



# Summary

---

- Framework design and application design (separation of modules) has big impact on performance
  - Bean code execution time is small
  - Fine-grained entity bean access does not scale
  - Optimization on remote/local communication, container design all have impact on performance.

# Problems

---

- Impact of different performance of backend database server
- Tuning of performance parameters?
  - E.g. TomCat worker thread number.