

Application Frameworks before System Frameworks

An OOPSLA 2000 Practitioner's Report

Jon Hancock -
jhancock@patternware.com

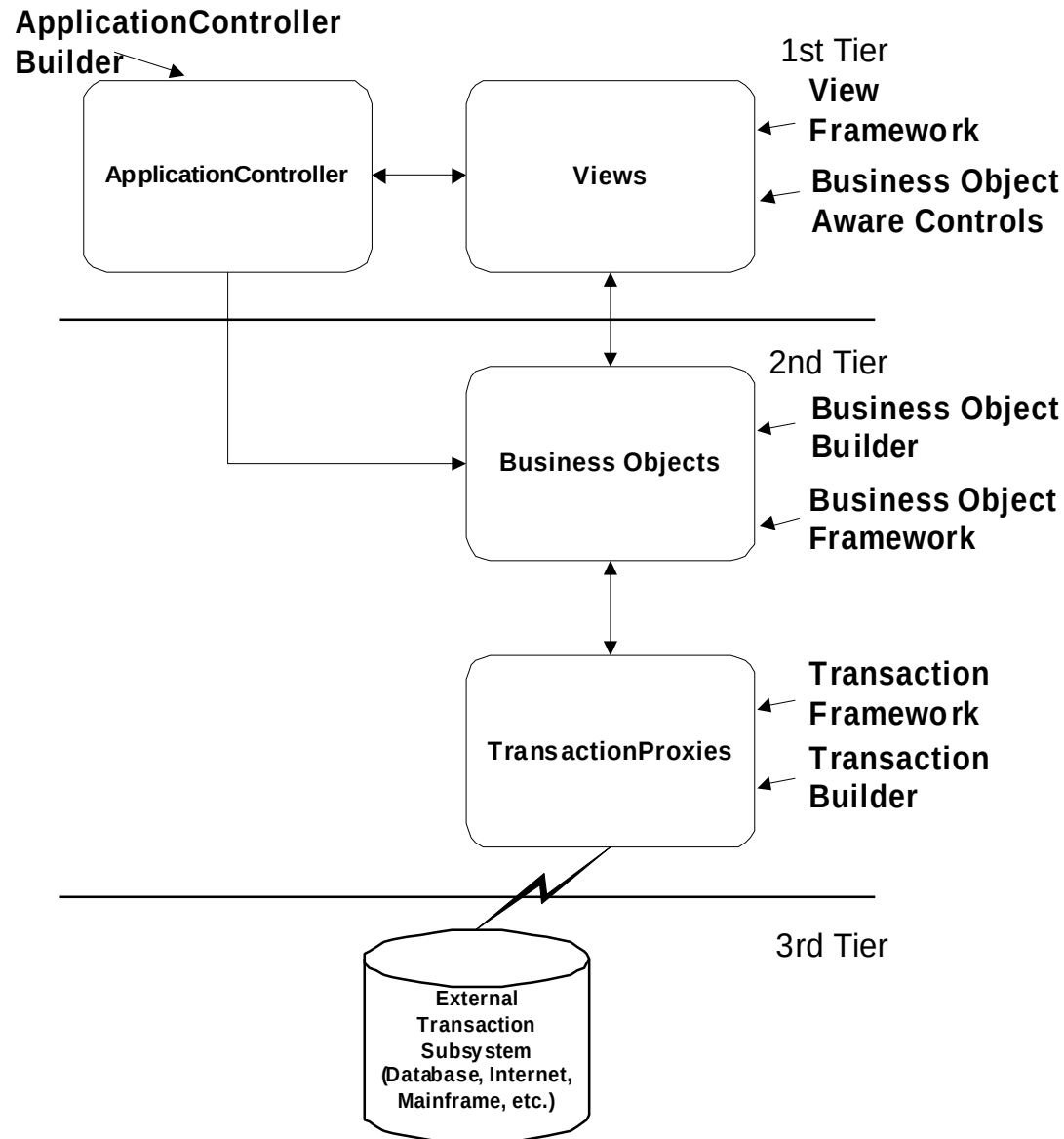
App Development Goals

- Predictable development cycles/timelines
- Maintainable code
- Enable **average programmers** to develop using a sophisticated architecture.
- Enable rapid iterative design/development cycles

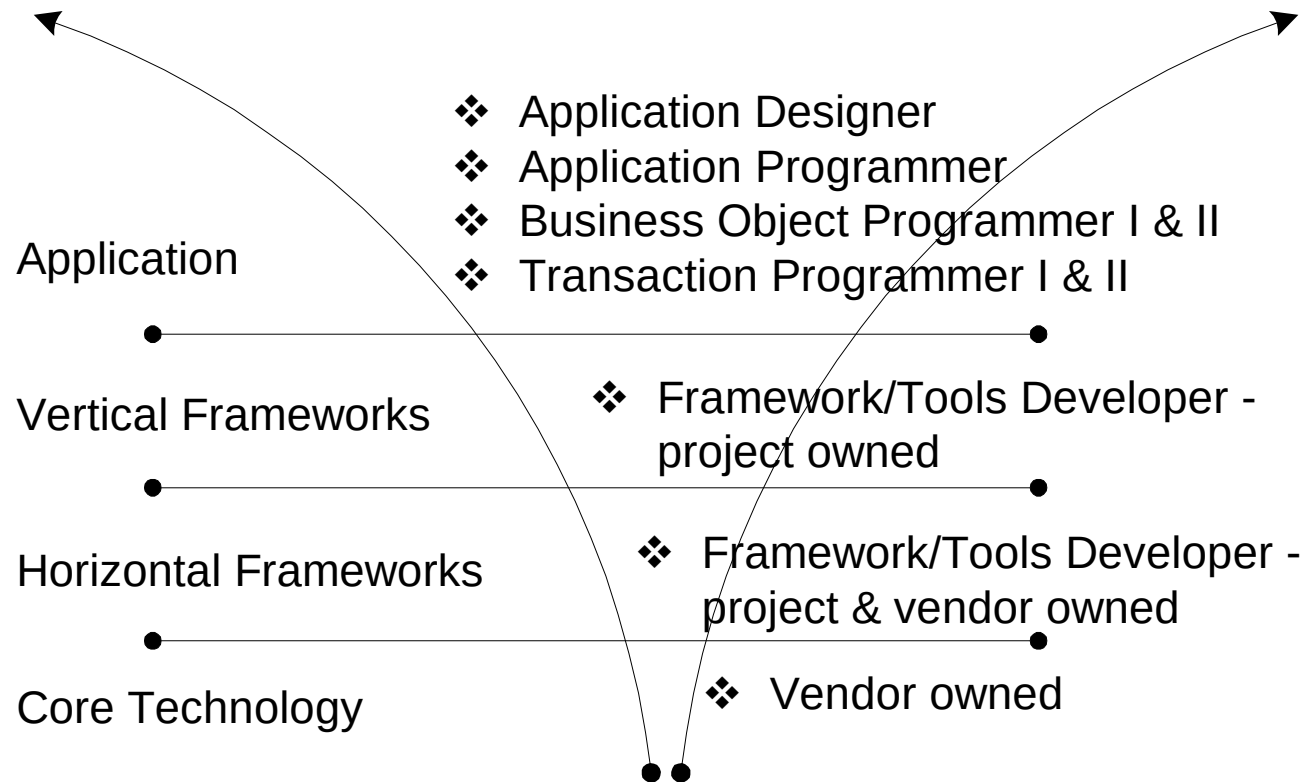
App Development Goals

- Enable design skills and most code to work across different system architectures
- Let the architects own the architecture, not the system vendors

High-Level Logical Model



Enable Training and Development by Role



Role - Application Designer

- No Java Knowledge Required
- All Tools Available
 - Training time: 2 weeks
- Average Object Design Knowledge
 - Training time: 1 - 6 months
- Good User Interface Design
 - Training time: 1 - 3 months
 - Good taste: innate

Role - Application Programmer

- Some Java Knowledge
 - Training time: 1 - 6 months
- Use Tools
 - Training time: 2 weeks
- Application Design Knowledge
 - Training time: 1 - 6 months in parallel with Java

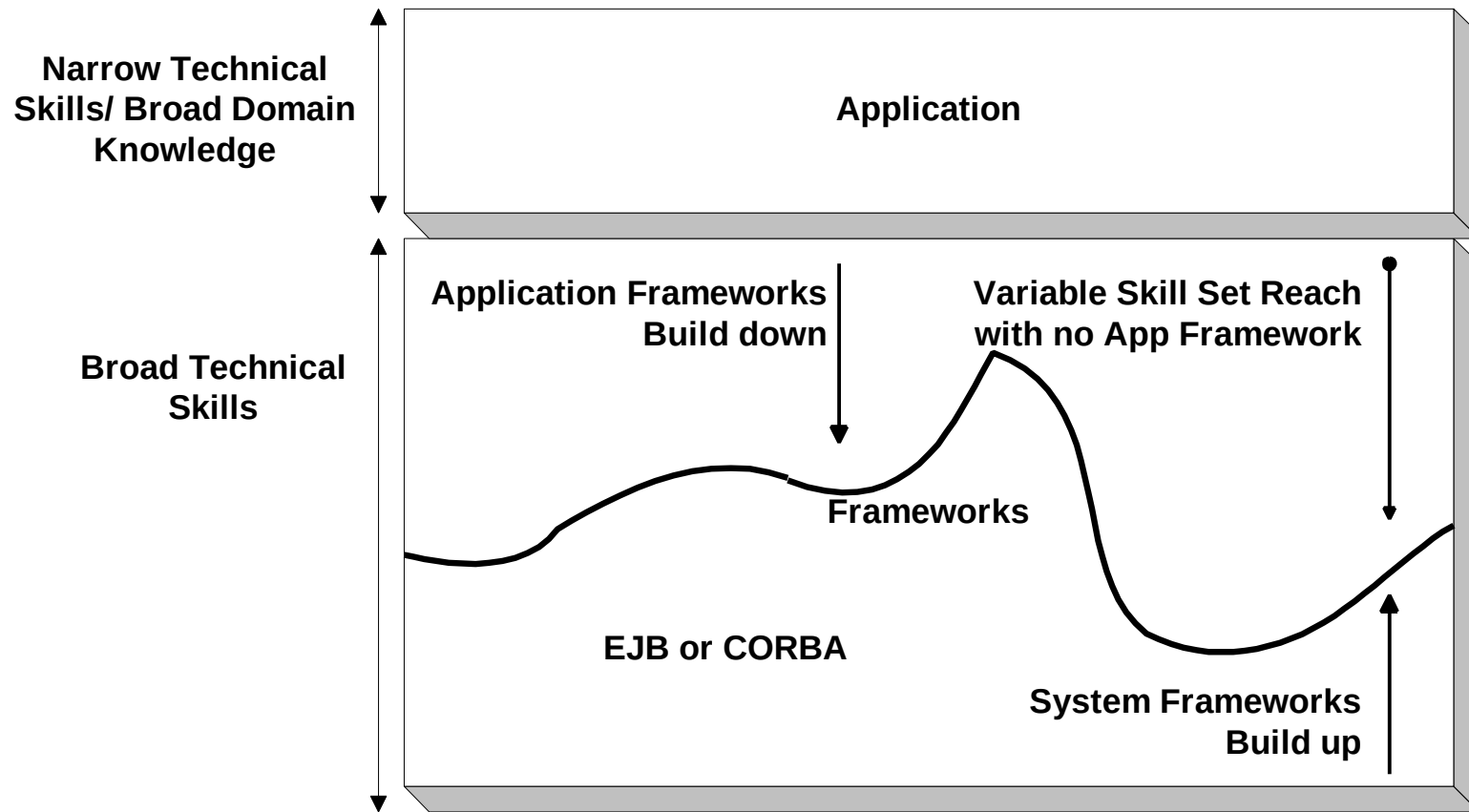
Role - Business Object Developer

- Level I - Beginner
 - Little Java knowledge
 - Object design knowledge: 1 - 3 months
 - Similar skills as Application Designer
- Level II - Advanced
 - More difficult business rules
 - Solid programming background
 - 1+ years of Java/Object experience

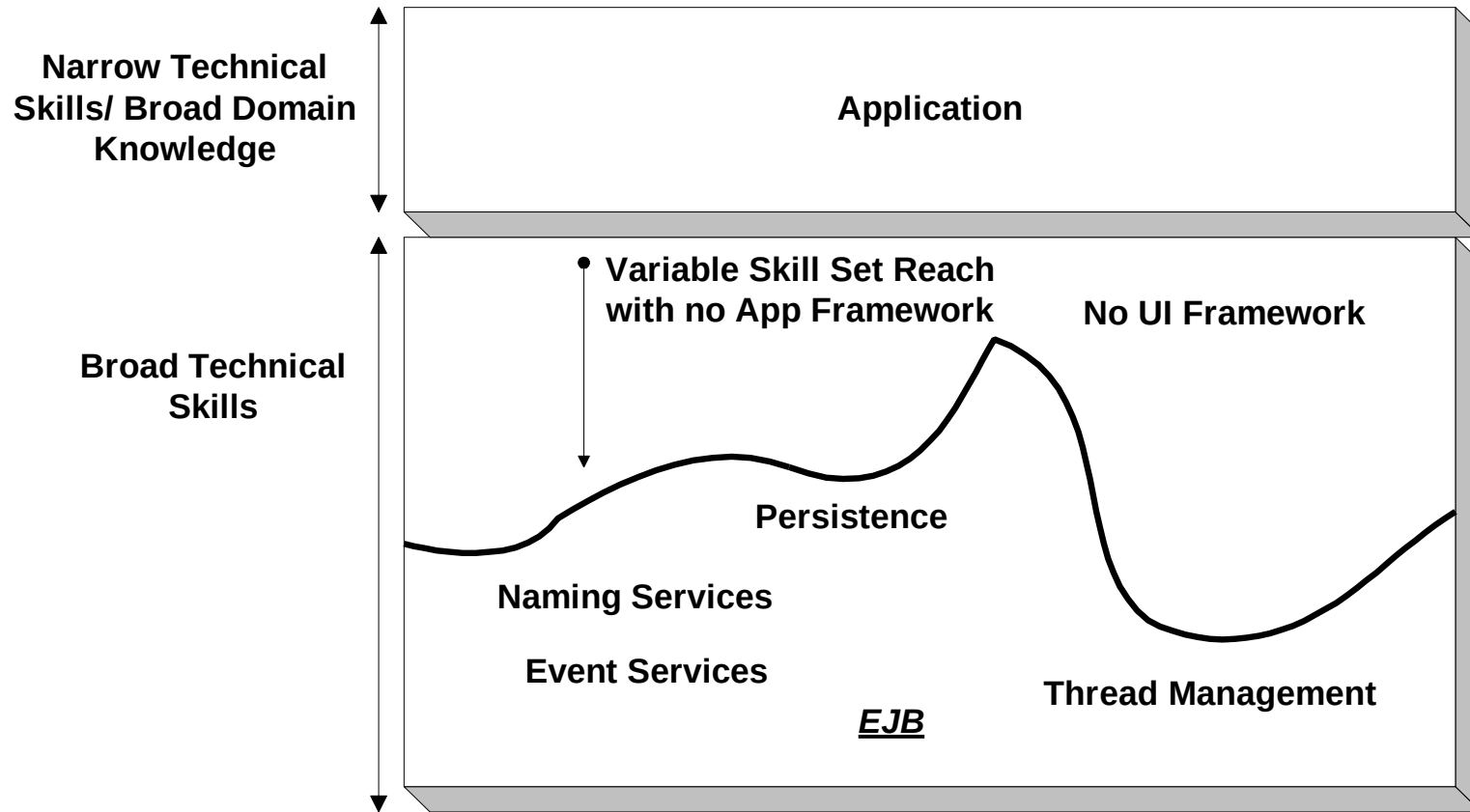
Role - Transaction Developer

- Level I - Beginner
 - Relational data design knowledge: 2 - 6 months
 - Use tools to develop: 1 month
- Level II - Advanced
 - Strong data design: 1+ years
 - Solid programming background
 - 1+ years of Java/Object experience

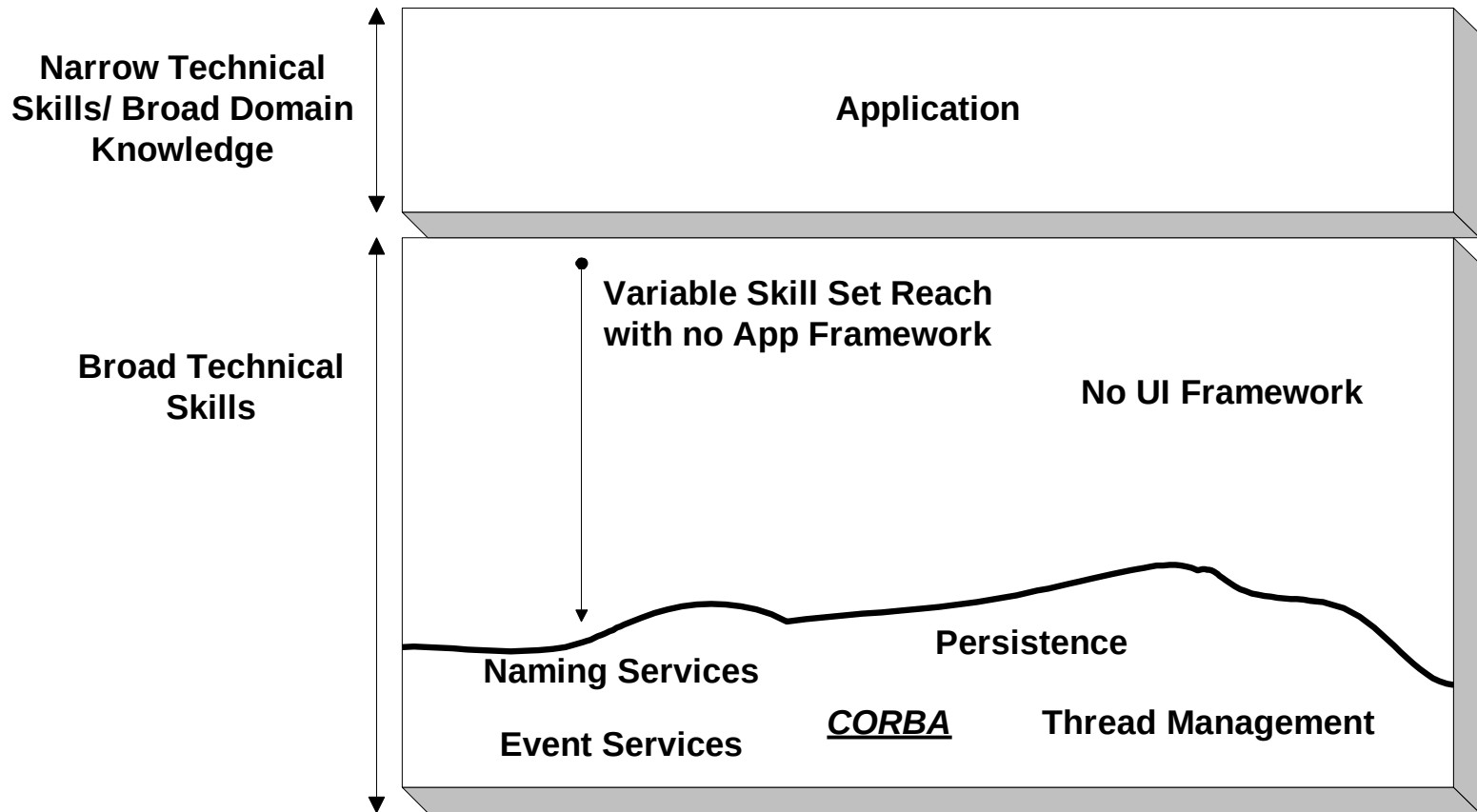
App vs. System Frameworks



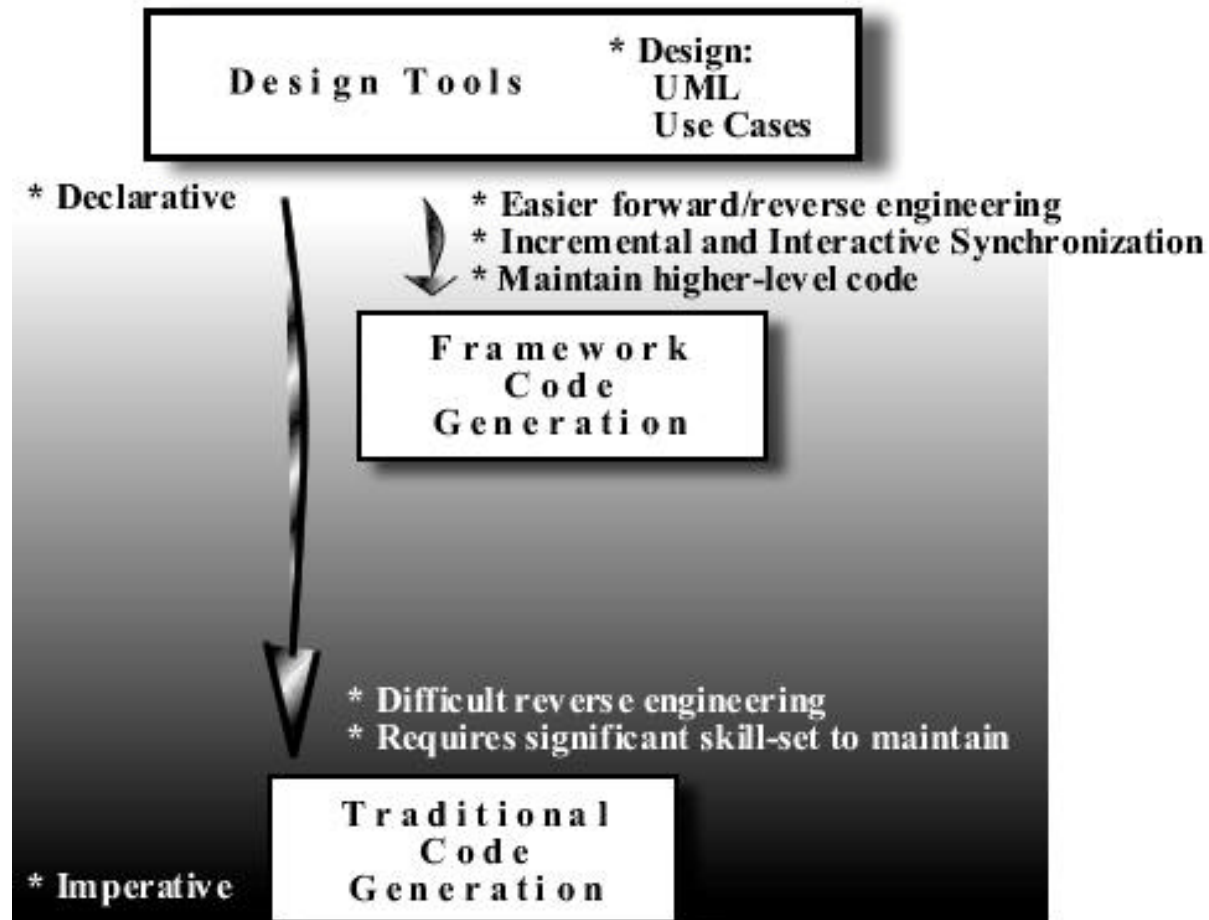
App vs. System Frameworks



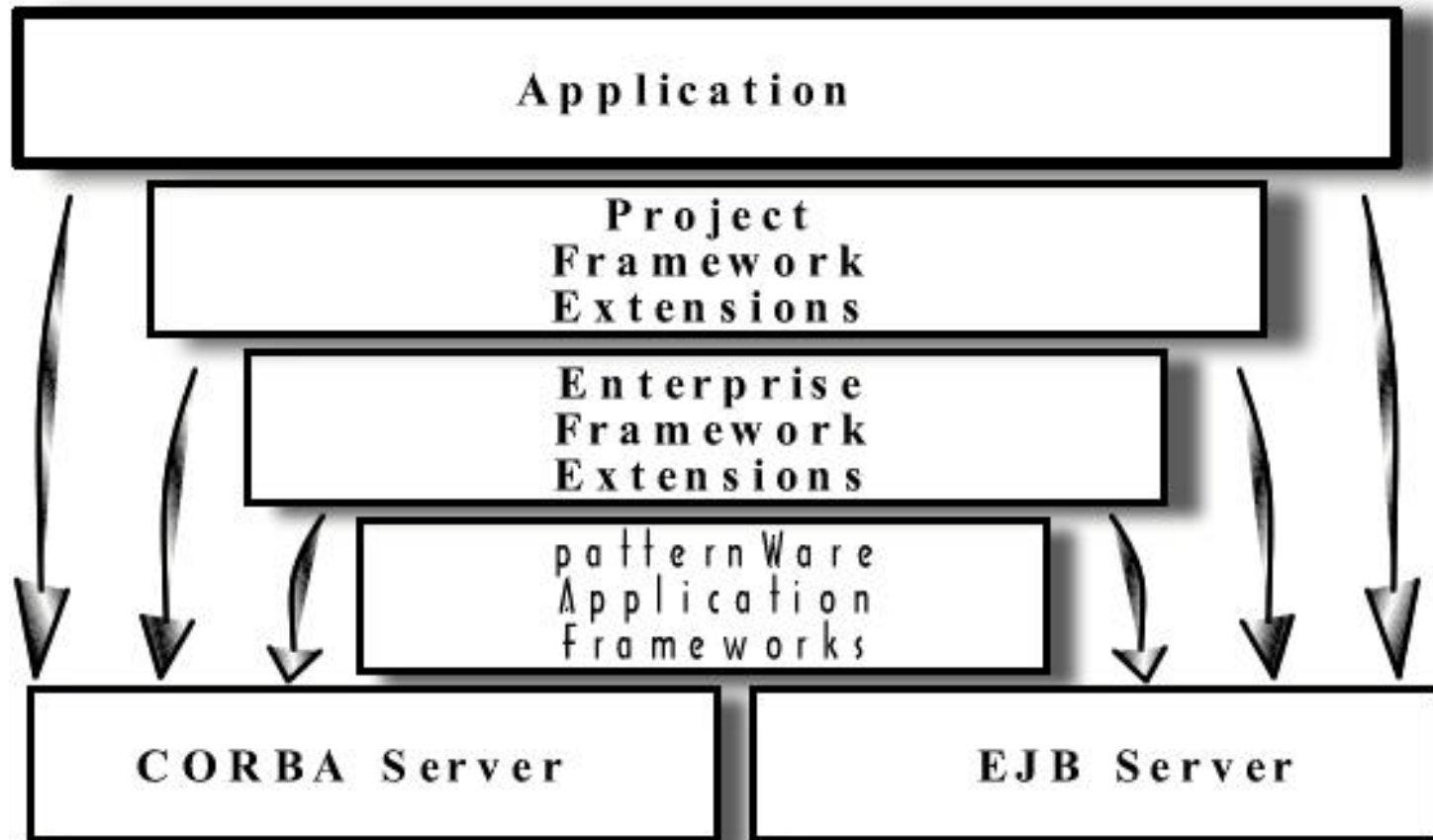
App vs. System Frameworks



Skill Set Reach Litmus Test



Intended to be Extended



- Lightweight Application Frameworks
- Provide services for most common application patterns
- Discourage but never preclude bypassing the Frameworks

Case 1: System Scope

- Replace large COBOL app for financial and organization management.
- Must be deployable on CORBA System Framework.
- Ensure cost-contained vendor independence.
- Traditional Client/Server with Internet capabilities

Case 1: Results

- Team - 2 COBOL programmers,
1 new programmer, 1 VB programmer,
1 DBA, and 2 Functional Analysts
- Time - 2.5 months. 1 Month training and design
- Results - 191 Views, 144 Business Object Classes
mapping to 93 Relational Tables.

Case 1: Results compared to CORBA Framework

- Greater than 50% code reduction
- Better client/server performance
and resource usage (5-10x object size)
- Significant error/bug reduction.
>80% reduction
- Predictable development cycles
- More maintainable code

Case 2: eunum Scope

www.eunum.com

- Next generation App Service Provider
- Must be able to repartition the app
close to deployment
- Intelligent but lightweight client download (<1MB)
- Must be embedded in a browser with
no Java runtime dependency

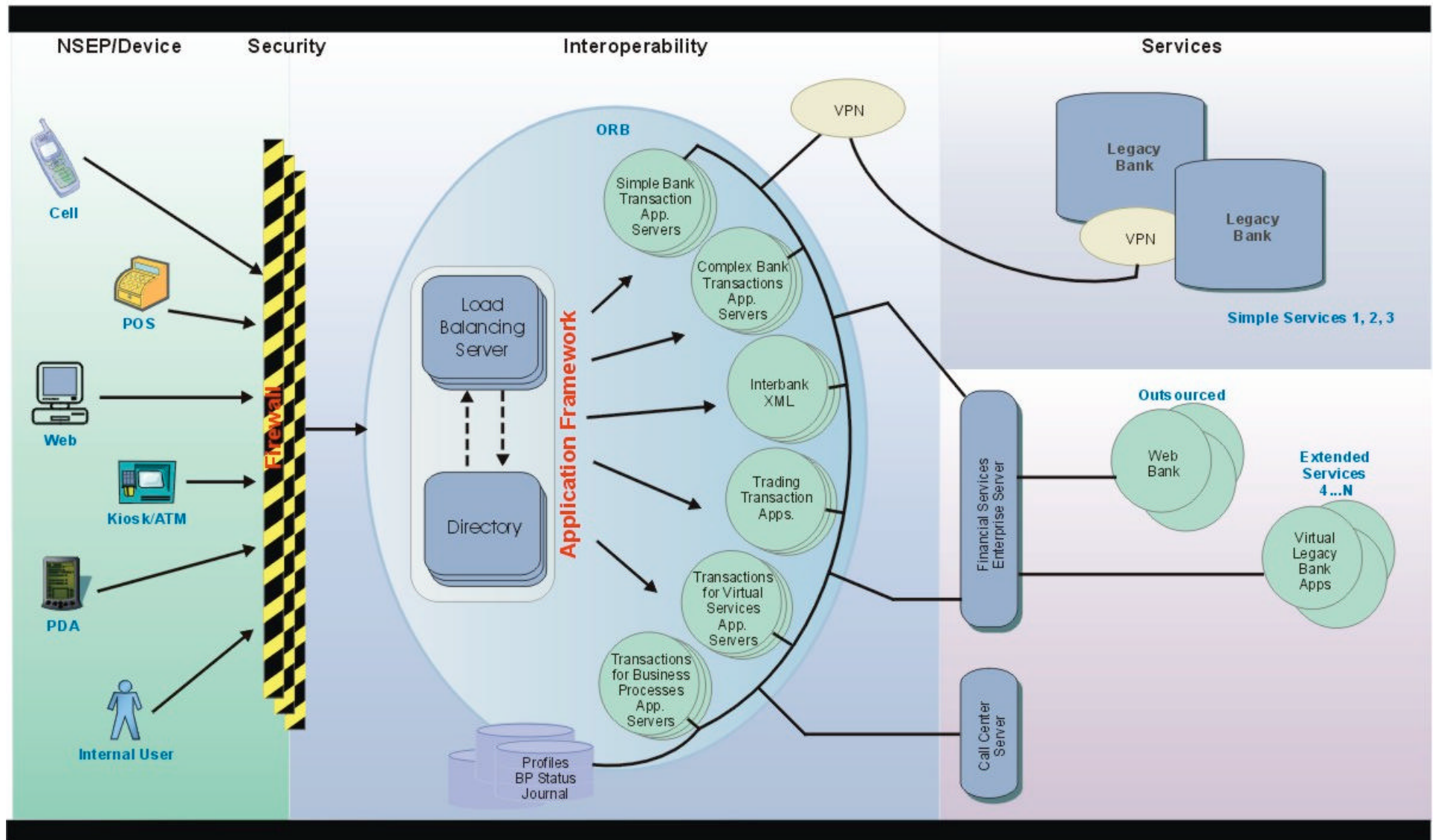
Case 2: eunum Results

- Current client download 1.1MB
- Blend of COM and Java on Client
- All Java on Server
- 10 - 100x bandwidth improvement over server side Web app

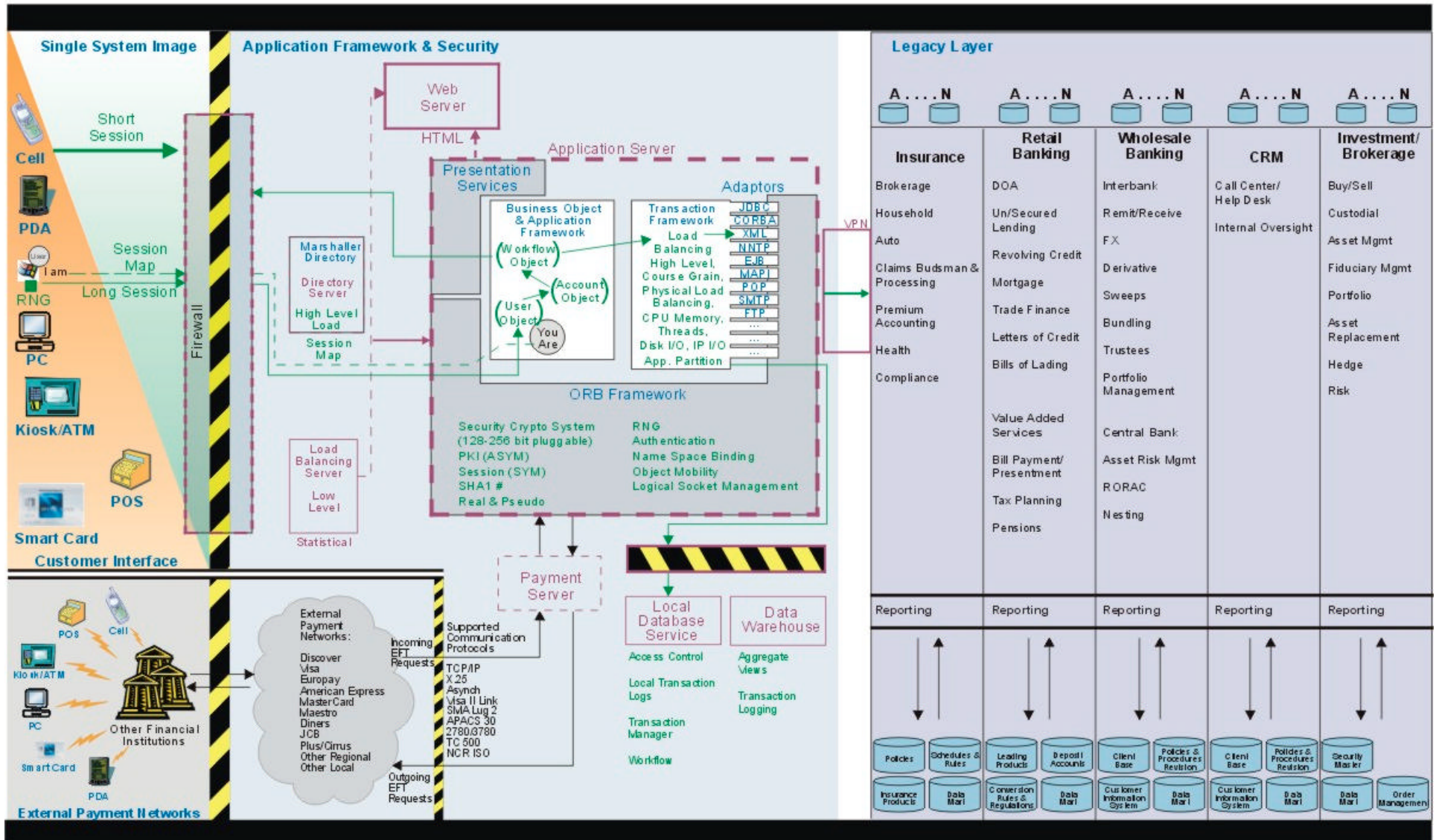
Case 3: e-Bank Scope

- Next generation Finance - App Service Provider for China
- Must achieve new levels of operations scalability
- No Java skills available
- Little Object modeling skills available

FWEC Logical Server Architecture



FWEC Customer Focused Architecture



Case 3: e-Bank Results

- First level of training finished
- Project is on track for e-Payment gateway integrating all China banks
- No System Framework has been chosen for deployment

Additional Lessons

- Not for the faint of heart
 - We now have over 600 classes in this suite of frameworks. ~5 MB of Java source.
- Did we build too much??
 - ORB
 - Thread Management
 - Platform Dependent UI Controls
 - JDK Replacements (Serialization, Reflection, Sockets, Collections, etc...)