Building Versatile Publish/Subscribe Infrastructures:
An Empirical Study of Middleware Versatility Approaches and their Trade-offs

Motivation

In order to stay current with the constant evolution of application and networking requirements, Middleware must be versatile. The development of versatile middleware, however, is non trivial. From the point of view of the infrastructure developers, it requires the right balance of software qualities such as configurability, extensibility, and maintainability; whereas from the point of view of the infrastructure users, it requires the right amount of performance, reusability and usability. Different **versatility approaches** have been adopted in the development of publish/subscribe infrastructures as follows:

1) **Minimal core systems** such as Siena and Scribe, that provide simple and generalized services.
2) **Coordination languages** as Linda, IBM TSpaces and JavaSpaces that provide a common vocabulary for the development of distributed applications.
3) **One-size-fits-all infrastructures** as CORBA-NS and READY that support a large and configurable set of features.

The lack of extensibility of these approaches motivated our development of YANCEES, a flexible (extensible and configurable) pub/sub infrastructure based on plug-ins and extensible languages.

Evaluation

This poster reports a quantitative and qualitative study that compares **YANCEES** with different open source publish/subscribe infrastructures (**Siena**, **JavaSpaces**, **CORBA-NS**) developed according to different versatility strategies above.

In our evaluation, we selected three **feature-rich event-driven applications from different domains**: EDEM for usability monitoring, CASSIUS awareness server and IMPROMPTU peer-to-peer file sharing.

We abstracted their publish/subscribe requirements in the form of **reference APIs**, implementing these APIs, using the selected publish/subscribe infrastructures, as shown in Figure1.

The resulting implementations were analyzed and compared with one another based on different software qualities: **reusability**, **modularity, flexibility, usability** and **performance**.

These software qualities were expressed in terms of code metrics including (Lines of Code (LOC), McCabe Cyclomatic Complexity (CC), Diffusion and Degree of Scattering of Concerns.

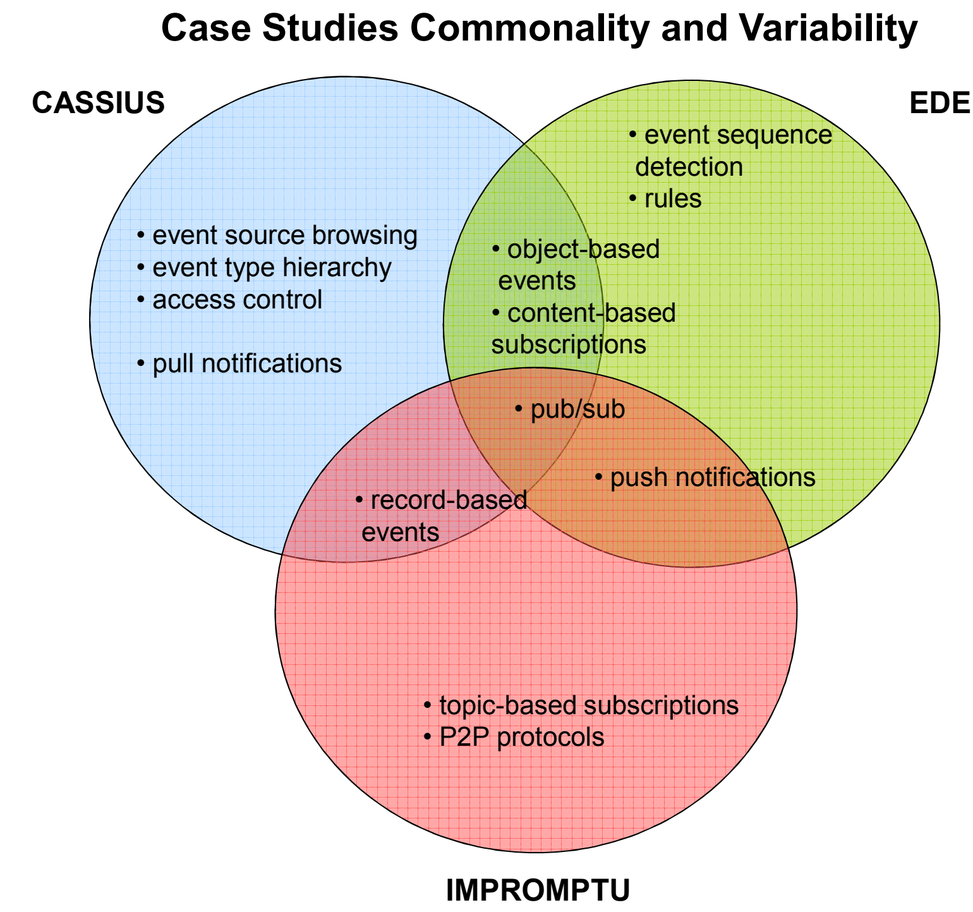
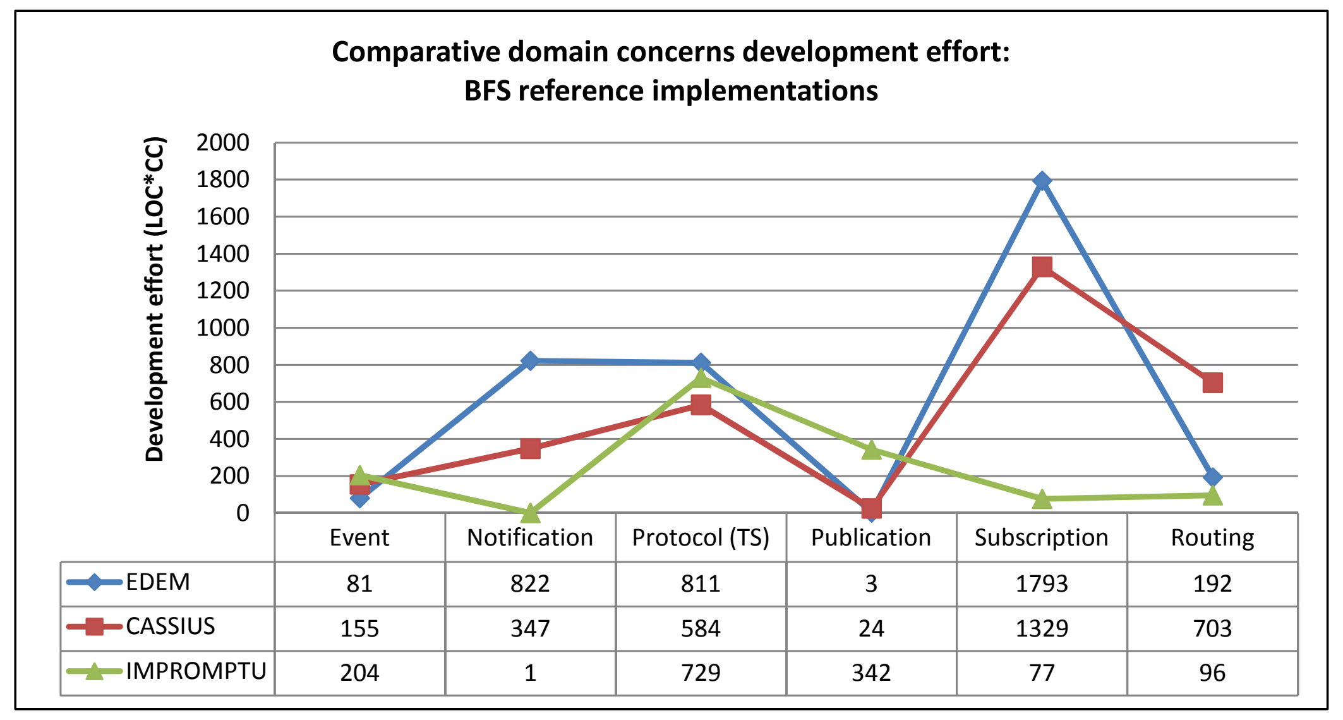


Figure 1: Case studies commonality and variability



	Event	Notification	Protocol (TS)	Publication	Subscription	Routing
EDEM	81	822	811	3	1793	192
CASSIUS	155	347	584	24	1329	703
IMPROMPTU	204	1	729	342	77	96

Figure 2: Three case studies concern sizes

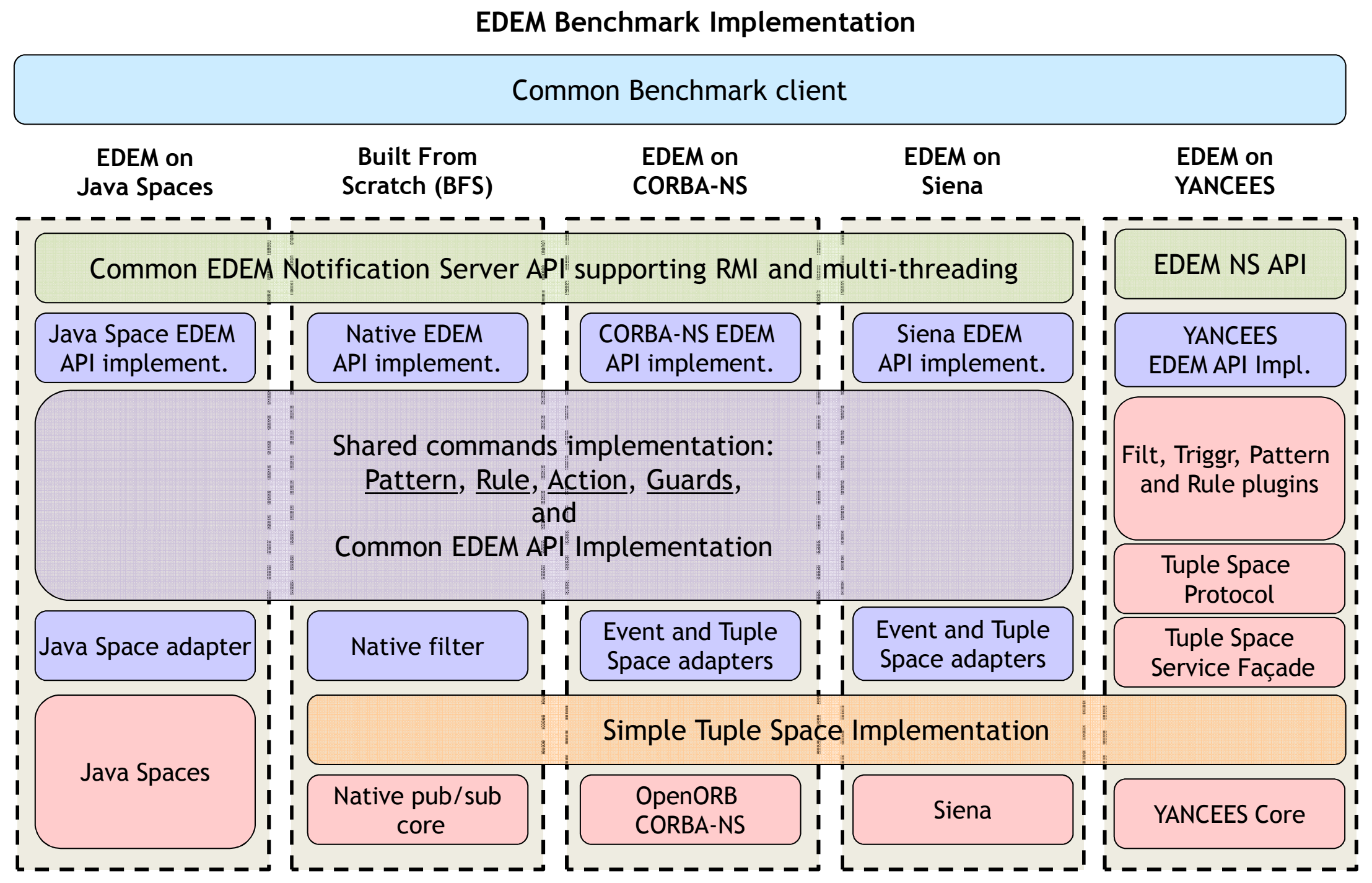

EDEM Benchmark Implementation
Common Benchmark client
EDEM on Java Spaces
Built From Scratch (BFS)
EDEM on CORBA-NS
EDEM on Siena
EDEM on YANCEES
Common EDEM Notification Server API supporting RMI and multi-threading
EDEM NS API
Java Space EDEM API implement.
Native EDEM API implement.
CORBA-NS EDEM API implement.
Siena EDEM API implement.
YANCEES EDEM API Impl.
Shared commands implementation: Pattern, Rule, Action, Guards, and Common EDEM API Implementation
Filt, Triggr, Pattern and Rule plugins
Tuple Space Protocol
Java Space adapter
Native filter
Event and Tuple Space adapters
Event and Tuple Space adapters
Tuple Space Service Façade
Java Spaces
Simple Tuple Space Implementation
Native pub/sub core
OpenORB CORBA-NS
Siena
YANCEES Core

Figure 3: EDEM case study architecture

Case Studies

• **EDEM** is a rule-based usability monitoring tool that looks for event patterns on streams of GUI events.
• **IMPROMPTU** is a peer-to-peer file sharing tool that relies on fast P2P topic-based event routing
• **CASSIUS** is an awareness server that supports content-based filtering and event source advertising and browsing.

Results

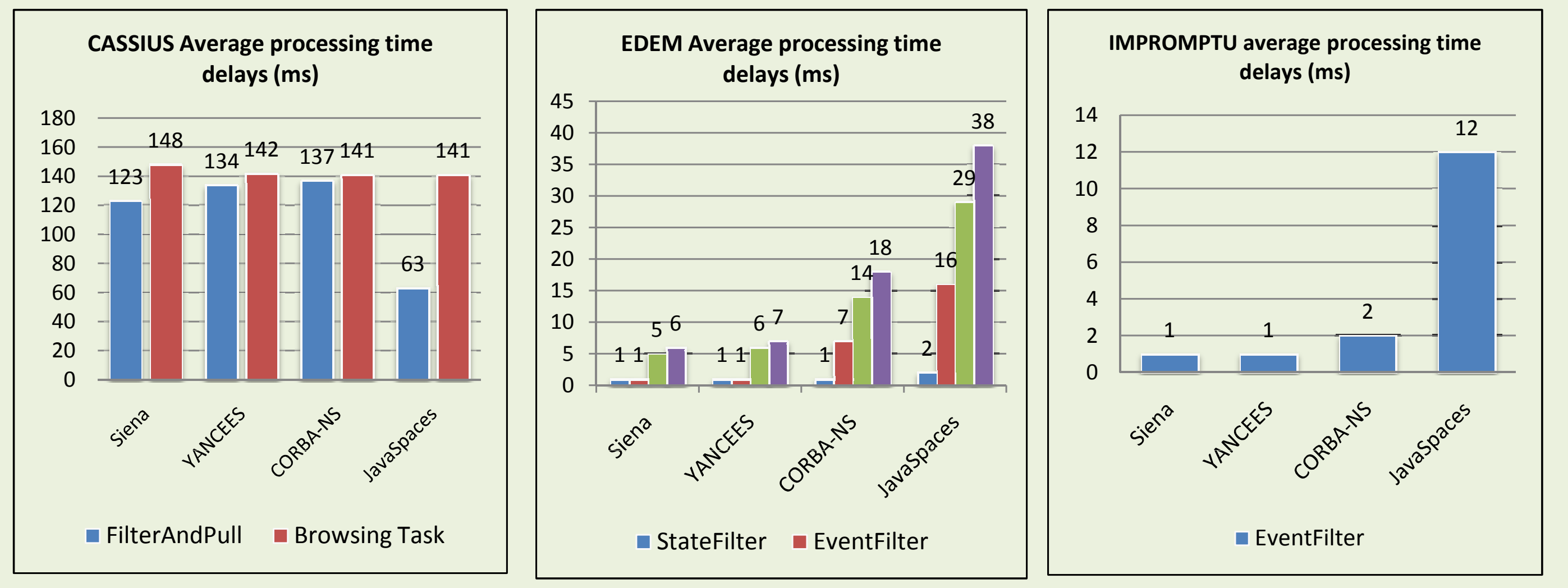
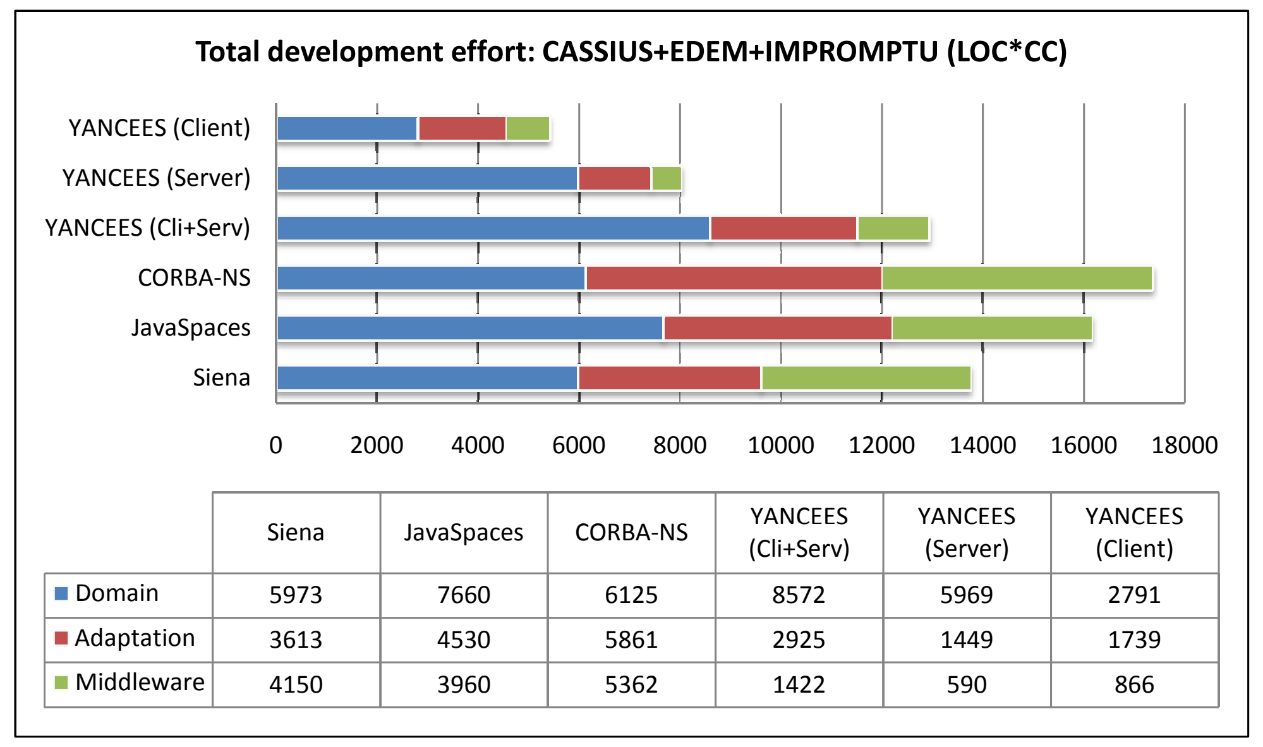
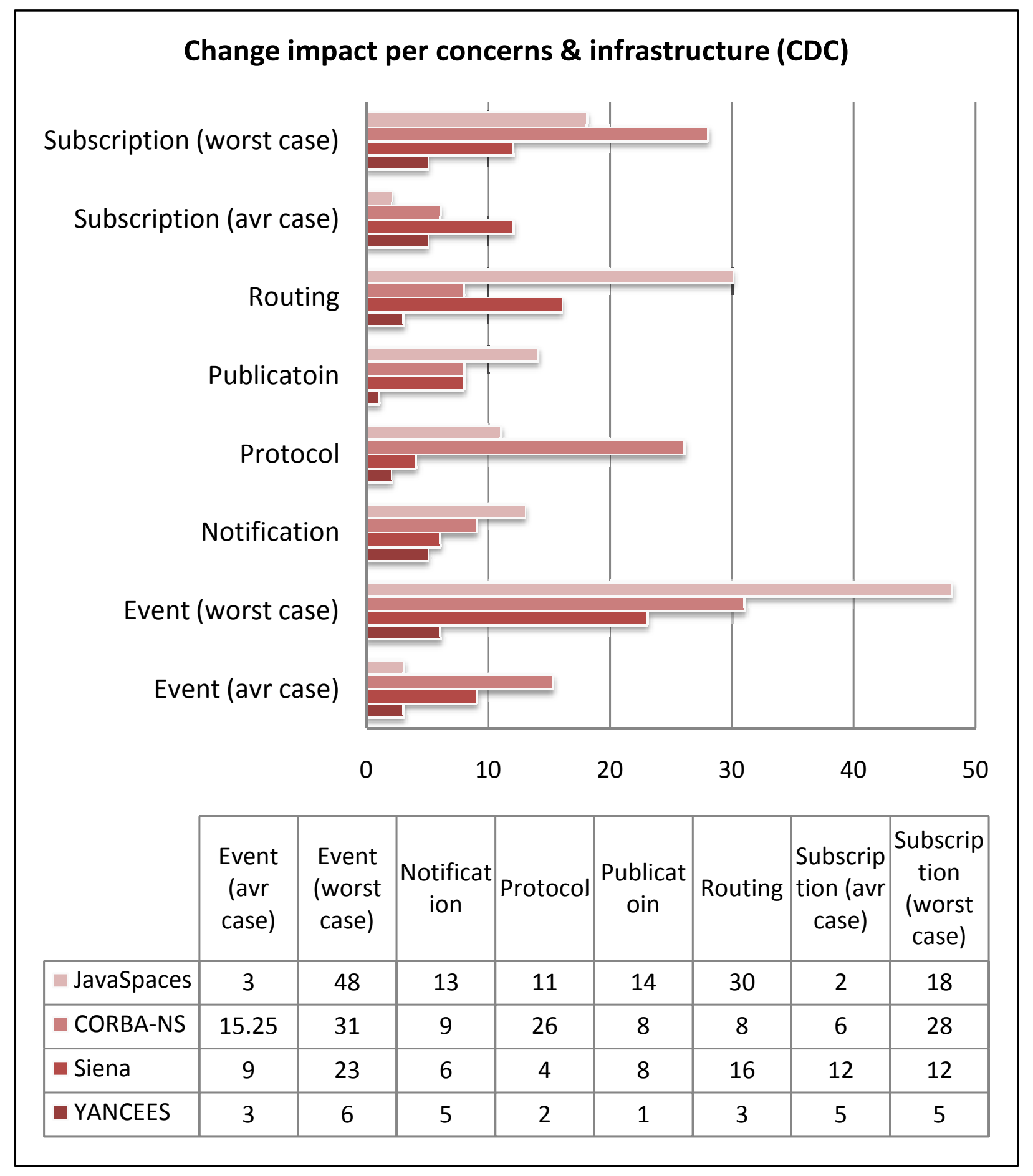


Figure 4: Performance results for the three case studies (smaller is better)



	Siena	JavaSpaces	CORBA-NS	YANCEES (Cli+Serv)	YANCEES (Server)	YANCEES (Client)
Domain	5973	7660	6125	8572	5969	2791
Adaptation	3613	4530	5861	2925	1449	1739
Middleware	4150	3960	5362	1422	590	866

Figure 7: Total development effort (smaller is better)



	Event (avr case)	Event (worst case)	Notification	Protocol	Publicatoin	Routing	Subscription (avr case)	Subscription (worst case)
JavaSpaces	3	48	13	11	14	30	2	18
CORBA-NS	15.25	31	9	26	8	8	6	28
Siena	9	23	6	4	8	16	12	12
YANCEES	3	6	5	2	1	3	5	5

Figure 5: Stability: change impact analysis (smaller is better)

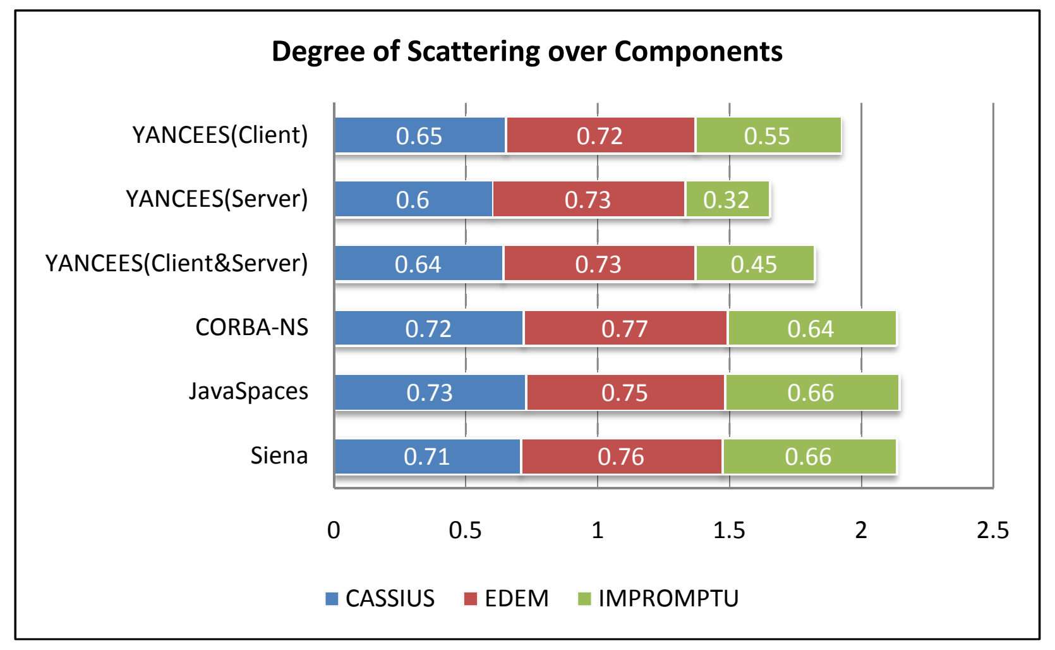
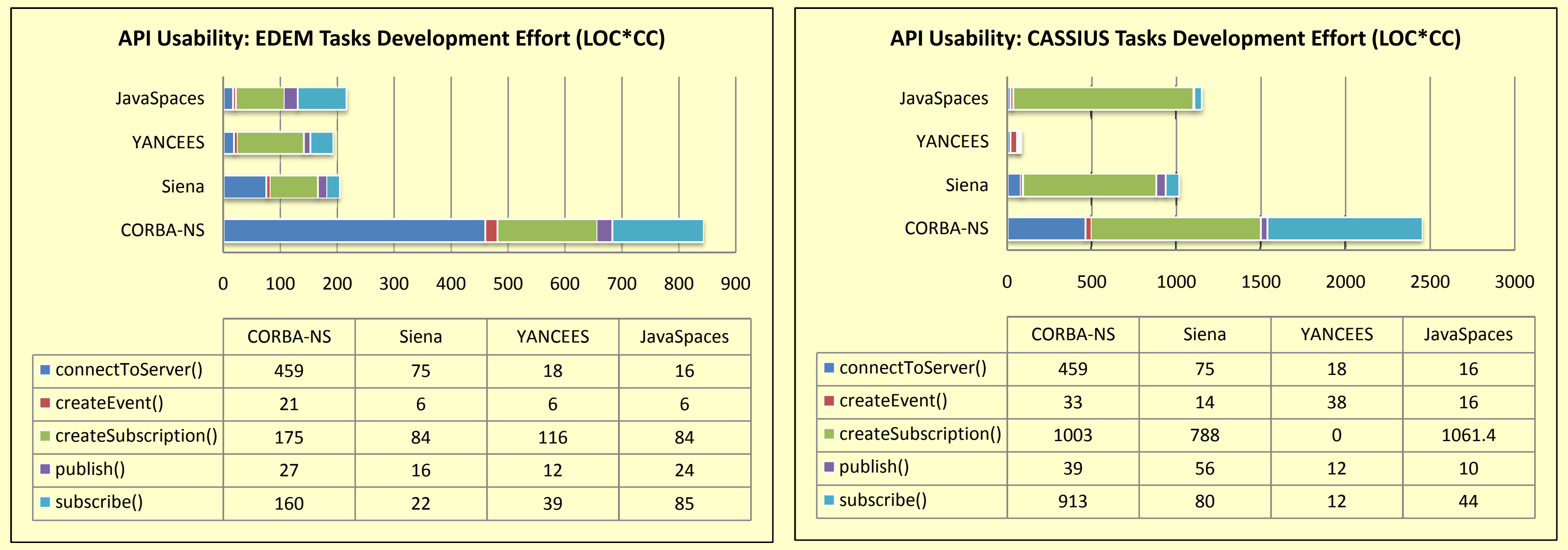


Figure 6: Implementation modularity (smaller is better)



	CORBA-NS	Siena	YANCEES	JavaSpaces
connectToServer()	459	75	18	16
createEvent()	21	6	6	6
createSubscription()	175	84	116	84
publish()	27	16	12	24
subscribe()	160	22	39	85

	CORBA-NS	Siena	YANCEES	JavaSpaces
connectToServer()	459	75	18	16
createEvent()	33	14	38	16
createSubscription()	1003	788	0	1061.4
publish()	39	56	12	10
subscribe()	913	80	12	44

Figure 8: API usability task analysis: Object-based (EDEM) vs. Text-based (CASSIUS) subscription case studies (smaller is better)

Findings

- The generality of minimal core infrastructures is effective and efficient in support of large variability of requirements and novel features, however, it is inflexible, leading to mismatches and adaptation costs.
- One-size-fits-all approaches, while support configurability, are slower, and have poor usability due to the many choices provided, and the mix of configuration and publish/subscribe APIs.
- Flexible infrastructures as YANCEES combine the configurability of one-size-fits-all approaches, with the performance and API simplicity of minimal core infrastructures (Figures 4 & 8) in a modular, more maintainable, implementation (Figure 6).
- Over successive reuses, YANCEES can reduce the development costs of building application-specific infrastructures (Figure 7).
- By supporting application-specific subscription languages, YANCEES reduces the abstraction distance between required (application domain) and provided (infrastructure) features.

Contact Information
Professor David F. Redmiles
Roberto S. Silva Filho
Institute for Software Research
University of California
Irvine, California 92697-3425
{redmiles, rsilvafi}@ics.uci.edu

To learn more about YANCEES and have access to a prototype of the system and documentation, please visit the website:
http://isr.uci.edu/projects/yancees

This research was supported by the U.S. National Science Foundation under grant numbers 0534775, 0205724 and 0326105, an IBM Eclipse Technology Exchange Grant, and by the Intel Corporation.

June 2009