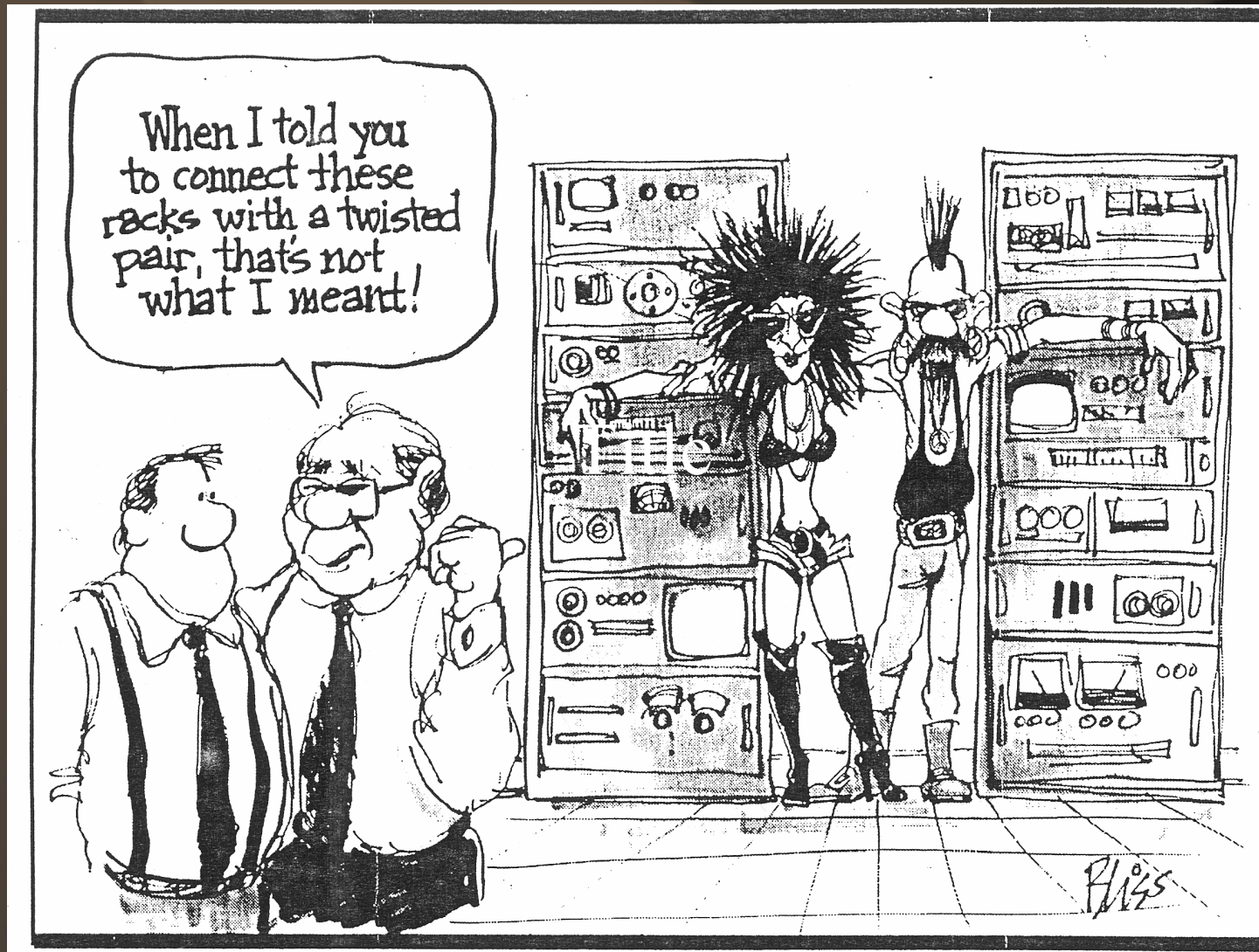


New Challenges to Quality in the 24x7 Enterprise IT Shop: Post-Integrated Business Systems



Assertions

Skillset expectations for design and operations professionals have traditionally been different

SW technology changes are placing new demands on the 24x7 I.T. shop

These changes offer operations professionals and businesses new opportunities to excel, but not for free

Traditionally different expectations for Computer Design professionals and Computer Operation professionals

Mid-level design engineer

- Bachelors degree in computer science, computer engineering or electrical engineering required; Masters degree highly preferred
- 7-10 years experience designing, building and maintaining complex information systems
- High levels of proficiency with the following technologies: UNIX/Linux, Java, EJB, SQL, C/C++, operating system internals
- Prior demonstrated success working as part of an engineering team

Mid-level operations engineer

- Bachelors degree desired; high school diploma with equivalent experience may be substituted
- 5 years experience in the operation and maintenance of both Windows and Linux systems
- Familiarity with concepts and usage of the following types of systems: ERP, accounting, human resources, data warehousing
- Familiarity with shell scripting and writing administrative tools in perl

The Traditional Approach: Pre-Integrated Systems

Requirements analysis performed by external consultants, often potential vendors

“Single-vendor” approach, to ensure everything works together

Vendor-restricted extensibility and operational interfaces; lack of operational options

Four Fundamental Changes are Leading to a New Era of Post-Integrated Systems

Service Oriented Architectures (SOAs)

Focus on providing Services instead of
Systems

Software As A Service (SaaS) delivery
mechanisms

Free Open Source Software (FOSS)

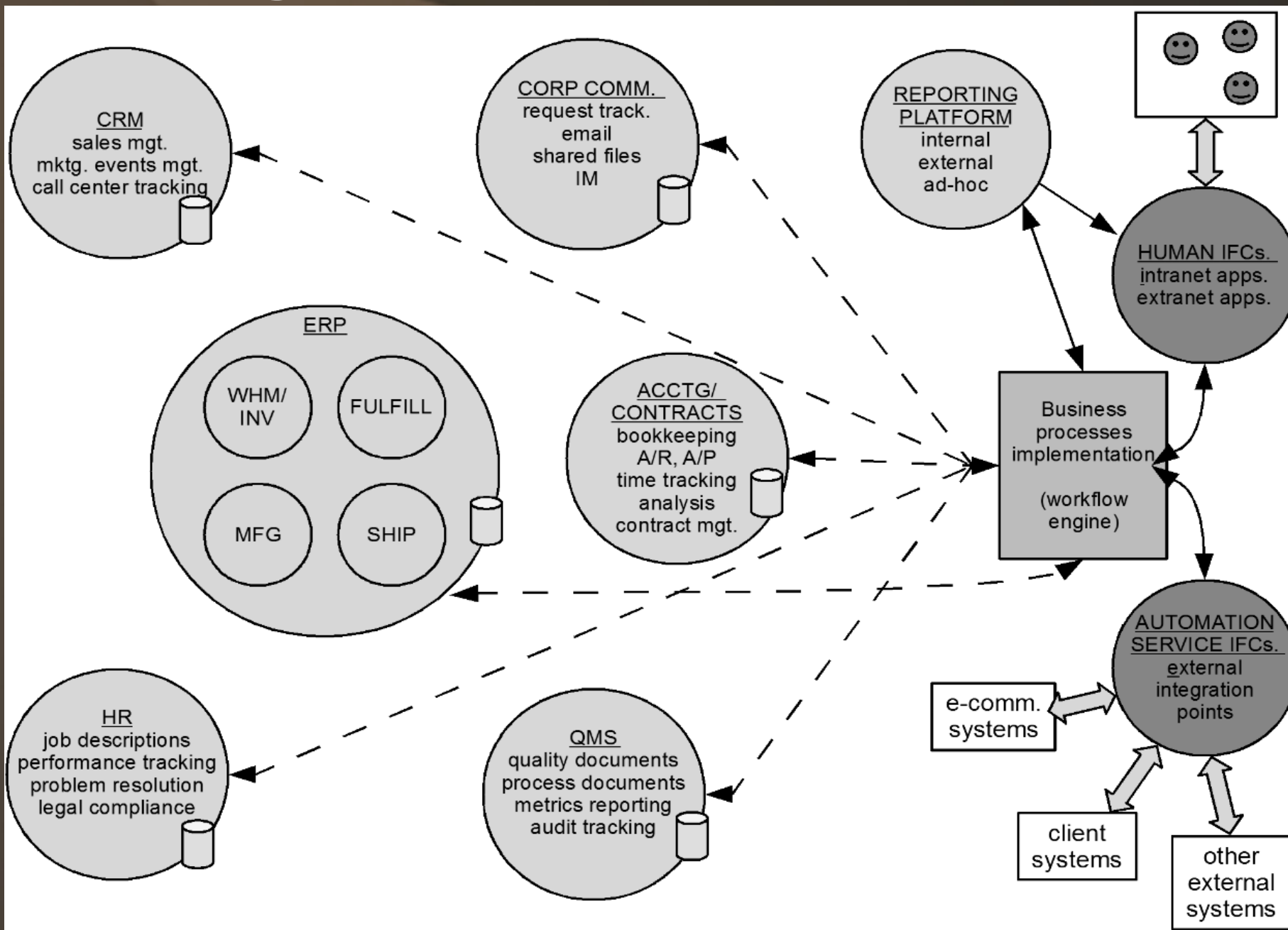
Impact of Service-Oriented Architectures (SOAs)

Can easily combine functional primitives across enterprise systems

Solutions can match business processes, instead of warping business processes to match built-in systems functionality

Enables true cross-vendor integration that is reliable and cost-effective

Enterprise automation platform utilizing SOA integration between functional systems



Opportunities and Challenges of Service-Oriented Architectures (SOAs)

Requires long-term architectural strategy to be driven by I.T. shop, not vendor

Requires competency in many development methodology skills

Looks a lot like design engineering, borrow what is already known!

Impact of Focusing on Services instead of Systems

Allows I.T. shop and end users to focus on supporting business tasks rather than deploying technology

Enables differentiating between *core services* and *non-core services*

I.T. shop can make more appropriate choices for deployment and support

Example from Co-Operations, Inc.

A leading 3rd-party logistics & fulfillment company

Core services: warehouse, fulfillment and production operations; accounting

- ▶ run in-house

Non-core services: email; calendaring; document control; request tracking; website/webstore hosting, etc.

- ▶ migrated to SaaS-delivered solutions from Google, NetDocuments, Extraview, Crucial Web Hosting, GoDaddy

Opportunities and Challenges of Focusing on Services instead of Systems

Incremental development techniques are required to meet changing service needs

On-going service refinement will require standardized design engineering tools such as Unified Modeling Language (UML)

Looks a lot like design engineering, borrow what is already known!

Impact of Software as a Service (SaaS) delivery mechanisms

Need to understand near-term and long-term roadmaps of SaaS providers

Frequent requirement for “federated” integrations: Single Sign-On, cross-domain event triggers, heterogeneous database views

Requires sophisticated network engineering techniques to meet SLAs

Opportunities and Challenges of Software as a Service (SaaS) delivery mechanisms

Requires sophisticated network engineering and integration technology knowledge to provide seamless solutions

Can provide significant ROI enhancements, and “pay as you go” opportunities

Looks a lot like design engineering, borrow what is already known!

Impact of Free Open-Source Software (FOSS)

Culture changes, including “no one to call when it breaks” concern

FOSS solutions being forced on I.T., due to perception it has no associated cost

Different skill sets required to leverage FOSS solutions, but the results can be worth the work

Opportunities and Challenges of Free Open-Source Software (FOSS)

Can require the full range of design/development skills and methodologies, must participate in FOSS ecosystem

Best opportunity to leverage enterprise advantage for funding skillset development

Looks a lot like design engineering, borrow what is already known!

What does this all mean?

Operations professionals are in a tough spot, but have significant opportunities

Must now justify services, not systems

ROI analyses must now include skillset improvement/maintenance

I.T. leadership must change culture(s)

What has to change?

Academic CS foundations will be required going forward

Work to define at least one new role:
I.T. Integration Engineer

Successful enterprises will have adjusted salaries

Shed past expectations

Three specific suggestions

Avoid the Big Bang (use incremental approaches)

Integration is the other 90% of the work, focus on it first

Design it like you plan to deploy it

Avoid the Big Bang

(utilize incremental approaches)

Abandon the “figure it all out before doing anything” approach

Study incremental approaches: Barry Boehm through modern Agile techniques

Use phased deployment to both meet business needs and refine requirements

Small increments, always user-ready

*Integration is the other 90% of the work
(focus on it first)*

Abandon the “functionality first, integration later” approach

Modern solutions delivery end-user functionality through primitives triggered across systems

Borrow “subsystem contract”, “interface-driven design” and “design for testability”

Design it like you plan to deploy it

Avoid the “what were these people thinking” legacy

Operations requirements are equal in importance to end-user requirements

Design for change: plan for systems to evolve over time, build in revision support

Explicitly plan for graceful degradation

*The Biggest Challenge
and
The Biggest Opportunity*

Remember that
Change Is Your Friend