

Business Rules Management and its relationship to Business Process Management, and Service Oriented Architecture

Introductory Lecture

Paul Vincent

Rules Specialist and Product Manager
Fair Isaac

$$= \sum_{s>s} \left\{ \frac{P_G f_G(s)}{f(s)} g - \frac{P_B f_B(s)}{f(s)} \right\} f(s) = [P_G(1 - F_G(s))g - P_B(1 - F_G(s))l]$$

Agenda

- ▶ Drivers for BRM, BPM and SOA
- ▶ The Business Rules Approach
- ▶ Rule Engines vs Rules Management
- ▶ Rules Management and IT
- ▶ Rule Management vs Process Management
- ▶ Q&A

POLL

▶ HOW MANY:

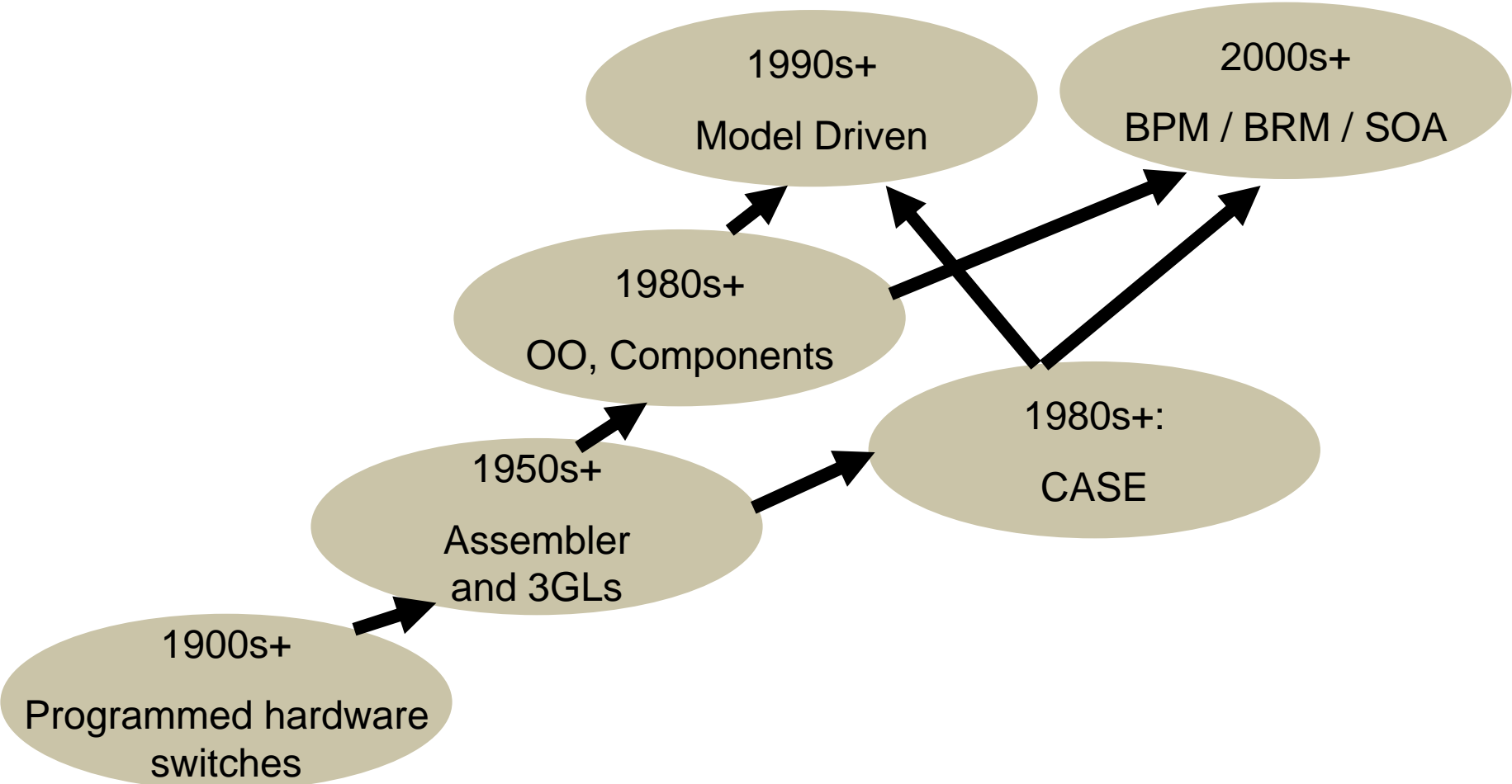
- ▶ Have (/are) implemented an SOA project?
- ▶ Have (/are) implemented a BPM project?
- ▶ Have (/are) implemented a business rules project?

Drivers for BRM, BPM, SOA...

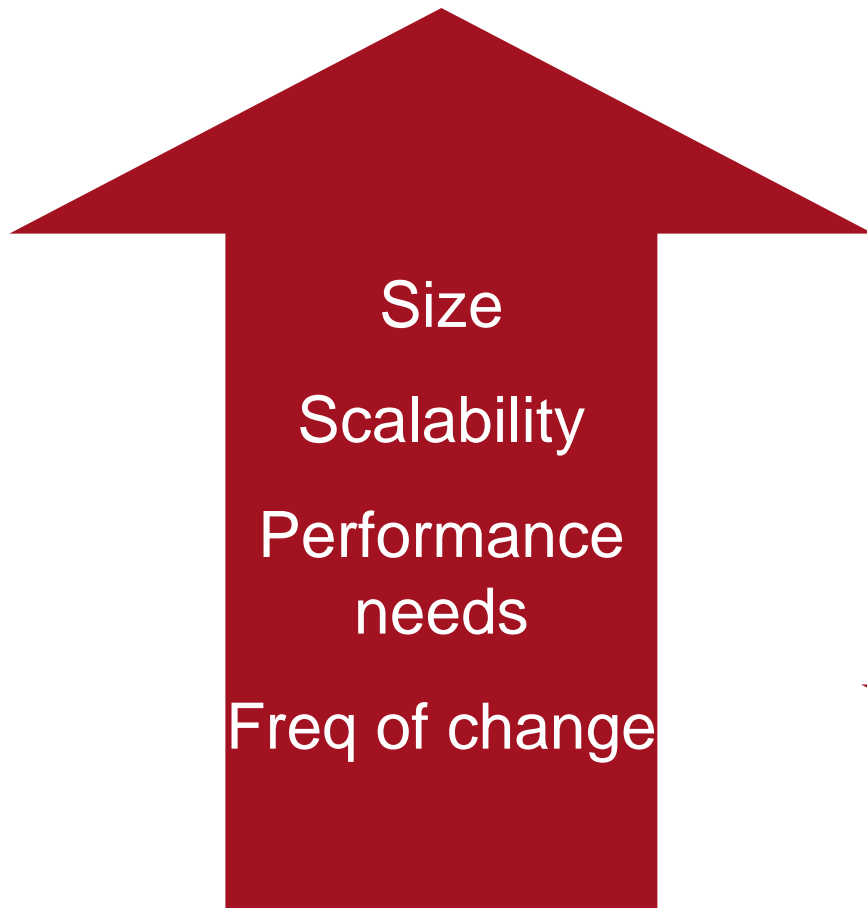
Why change?

History of Software development...

► ... is one of increasing abstraction

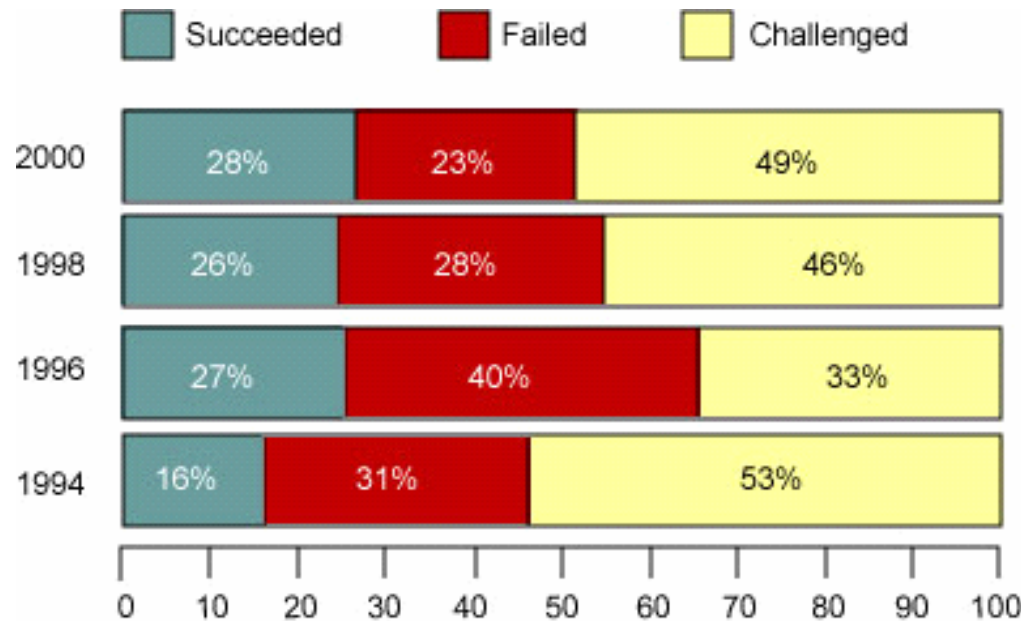


... for increasingly complex solutions



... and increasing problems

The high cost of IT project failure.



Nearly one quarter of IT projects fail, and nearly half are challenged in meeting their business goals.† The cost is staggering, estimated to run to \$350 billion per year.

† The Standish Group, CHAOS 2000 report.

Software development today



Software type (nb: not exclusive categories)	<u>Benefits</u>	<u>Costs</u>	<u>Examples</u>
Customised Solution	All-in-one, single vendor responsibility	Customisation, fit-to-need	ERP eg SAP
Tool and Component based	“Re-use”	System integration	RDBMS eg Oracle
Custom Application	Singular purpose, fit-to-need	Development costs, time	Many defence applications

Software development today: SOA, BRM, BPM

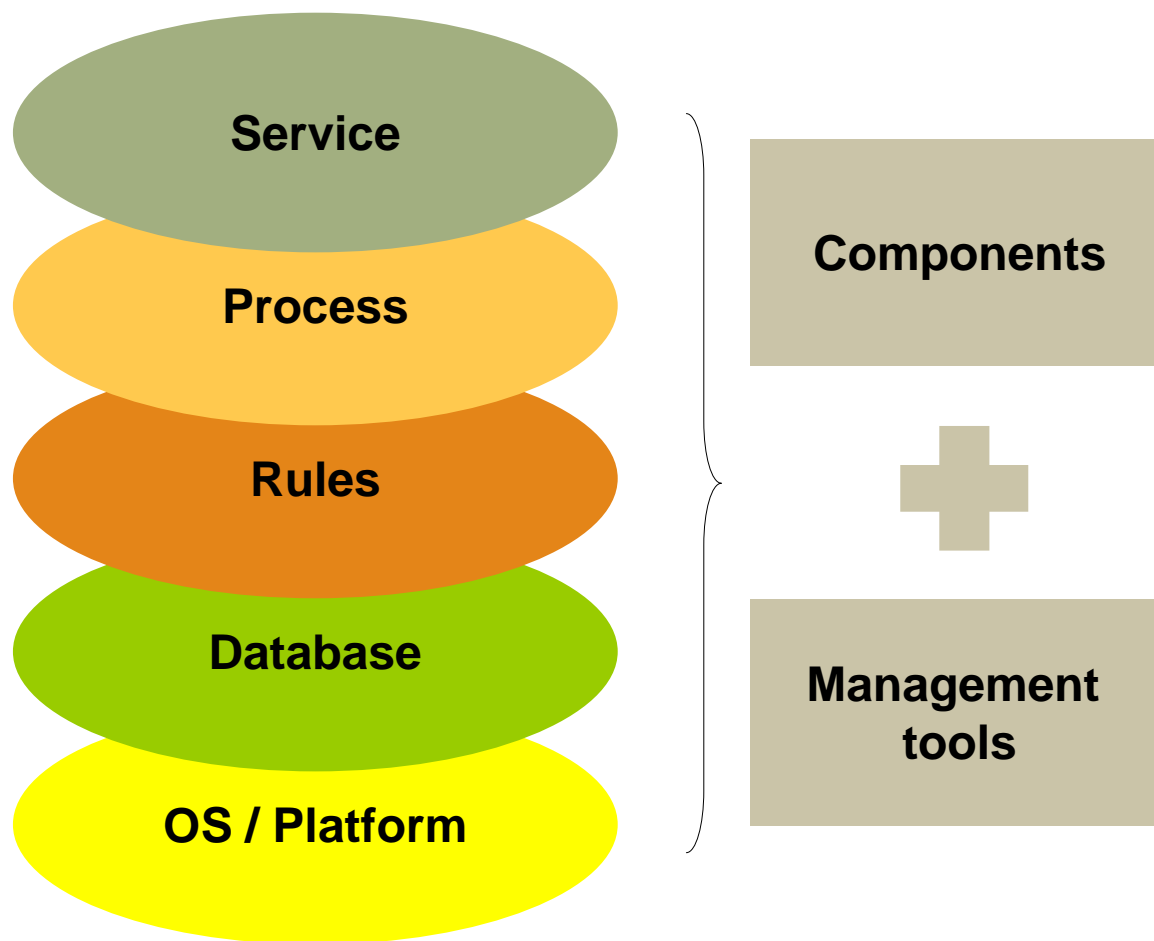
		<u>Costs</u>	<u>Examples</u>
SOA: services that can be re-used across an organization, and replaced as necessary to provide agility etc			
Custom Solution	All-in-one single vendor responsibility NOT APPLICABLE	Customisation fit-to-need NOT APPLICABLE	ERP eg SAP
Tools and Components	"Re-use"	System integration REDUCED	RDBMS eg Oracle
Custom Application	Singular purpose, fit-to-need	Development costs REDUCED	Many defence applications

BRM: business rules that represent the decisions made in applications can be maintained independently and at the business level

BPM: business applications can often be defined in terms of business processes, and managed as such

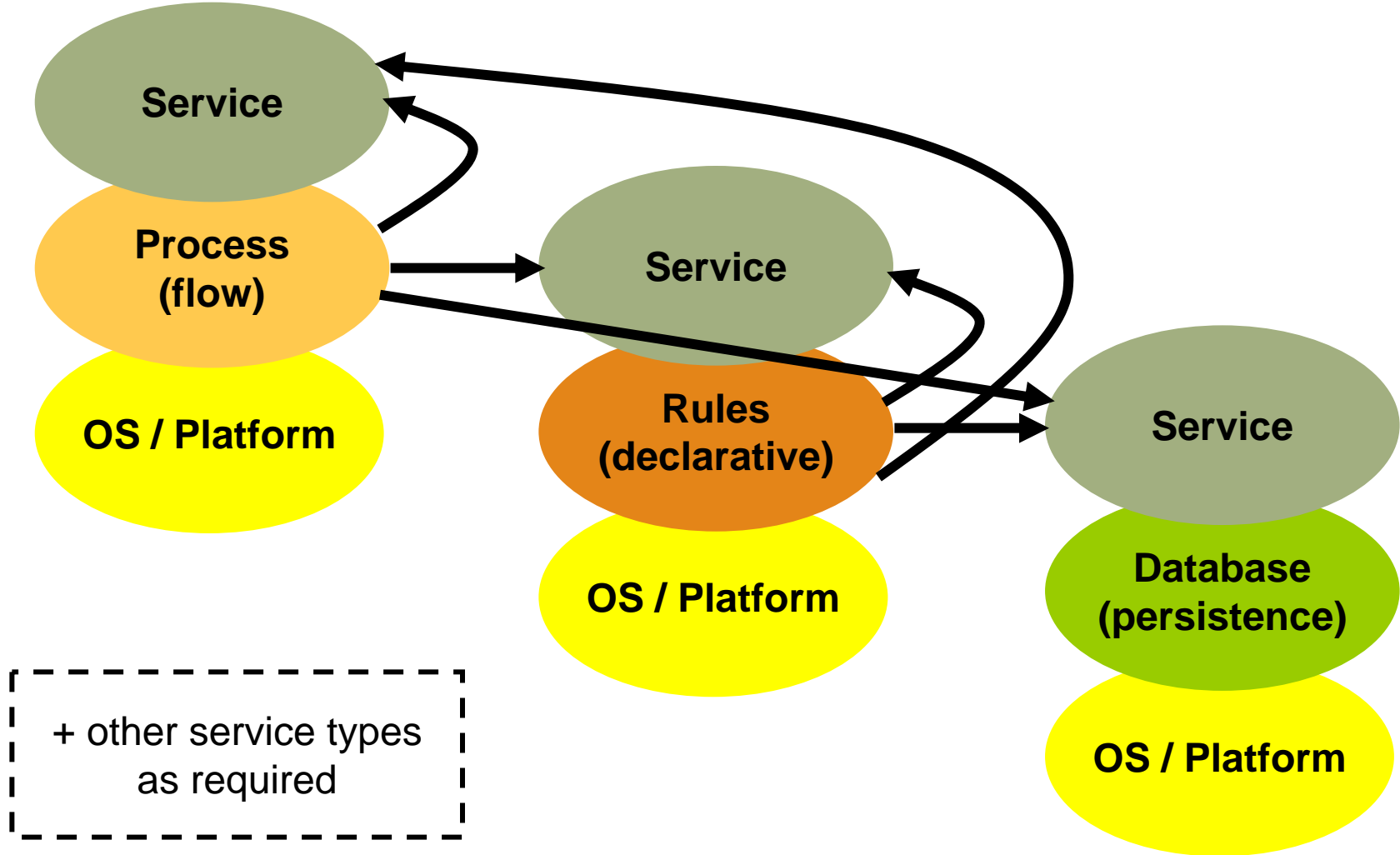
Software development today

- ▶ Recognition of a business IT stack:



Software development today

► Recognition of an optimal service-oriented IT architecture:



DISCLAIMER

- ▶ SOA = Service Oriented Architecture
 - ▶ An implementation of a service oriented approach
 - ▶ Use web services to implement the services
 - ▶ Uses SOAP as a messaging protocol
 - ▶ Uses BPEL for defining the service definition

- ▶ Service oriented approach != SOA
 - ▶ Services can be any software component mechanism
 - ▶ Web services considered by many to be an immature technology

What is “the Business Rules Approach”?

Business Rules Approach

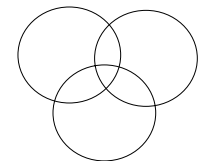
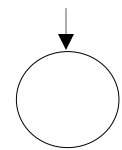
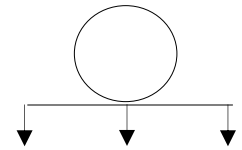
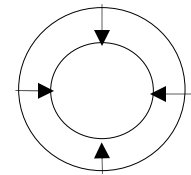
- ▶ Business rules should be defined, stored, reported, etc **separately** from other entities
 - ▶ In business documentation
 - ▶ In business process definitions
 - ▶ In use cases and system requirements
 - ▶ In code
- ▶ Business rules should be defined **declaratively**
 - ▶ Changing one rule definition should not require changes in another
 - ▶ Rules should be order-independent

Business Rules Approach

- ▶ Compare with “data management”
 - ▶ **Data** is defined and managed separately as a **common practice**
- ▶ Compare with “object orientation”
 - ▶ Objects should **encapsulate** data, behavior
 - ▶ ... but does not prevent us managing such data + rules separately

Main Drivers for using Business Rules Approach

1. Centralize / standard rule execution / management strategy in an organization
→ always know what rules are where
2. Apply rules in a standard way across channels / subsidiaries
→ always use the same rules from the same source
3. Allow businesses to control what rules are executed, and update them as required
→ return business decision control to the business and allow for timely changes to IT systems
4. Allow more complex processes to be automated
→ allow for planning, scheduling, best-choice type decisions to be made



Business Rules vs Business Processes

- ▶ Business processes consist of tasks (human or IT) sequenced in a flow
 - ▶ Comparable to visual flowchart programming
 - ▶ Simple to visualise, but large flows can be difficult to change
- ▶ Business rules define decision points and business logic
 - ▶ Comparable to business statements
 - ▶ Simple to visualise, but large rulesets can be difficult to navigate
- ▶ Typically rules are used to implement certain processes

POLL

▶ HOW MANY:

- ▶ Have business processes that are “naturally” rule driven?

“Rule Engines“ vs “Rules Management”

past and present

Rule based Programming Tools

Rule engines are specialised components for executing declarative rules

Component-based Rule engines
eg Blaze Advisor, JESS, JRules, Aion

OO Rule engines
eg Nexpert Object

Expert System shells
typically goal-driven

AI / KBS
development tools
eg ART, KEE

AI languages
eg LISP, PROLOG

Semantic web
& ontologies

1970s

1980s

1990s

2000s

How are Rule Engines evolving?

- ▶ **Rule engine** types:
 - ▶ Declarative (expressiveness) vs sequential (performance)
 - ▶ Forward & backward chaining and event-driven reasoning
 - ▶ Handling multiple rule types
- ▶ Rule execution **platforms**:
 - ▶ Java vs .NET vs C/C++/COBOL ...
 - ▶ Embedded device → PC → web services → mainframe
- ▶ Rule and policy **abstractions**:
 - ▶ Decision tables
 - ▶ Decision trees
 - ▶ Scorecards and score models

How are Rule Engines evolving?

- ▶ High **scalability & performance**:
 - ▶ Advances in rule execution algorithms
 - ▶ Rule servers for high multiples of transactions per second + simultaneous users
- ▶ Rule **expressiveness** to business users:
 - ▶ Rule syntaxes designed to be near Natural Language
- ▶ Multiple **data interface** capabilities:
 - ▶ 3GL (Javabean) support
 - ▶ Database and SQL support
 - ▶ XML support
 - ▶ Messaging and middleware support (CORBA, J2EE, MQ...)

How are Rule Engines evolving?

- ▶ Standards support: a good test of maturity:
 - ▶ Since 2002: **OMG** Business Rules Working Group (now Business Enterprise Integration task force)
 - ▶ Semantics for Business Vocabulary and Rules for formal business statement expressions
 - ▶ Production Rule Representation (est mid 2006) for if.. Then.. Rule interchange across software models
 - ▶ End 2005: **W3C**: Rule Interchange Format working group
 - ▶ **RuleML** standards body is often associated with W3C
 - ▶ **JSR**: JSR-94 Rule Engine Invocation standard for Java community

Rule standards are in progress

What is Rule Management?

- ▶ Rule engines only provide an **alternative** mechanism for implementing **behavioral rules** in software
 - ▶ Separation of rules from code
= conforms to Business Rules Approach
 - ▶ Rule language = specialist IT programmer required (sometimes)
- ▶ The main benefit from using a rule engine is the associated **rule management process**
 - ▶ Represent rules in near natural language
 - ▶ Rule reporting, verification, validation
 - ▶ Rule metadata, versioning, organisation by business function

1990s Problem:
rule representation & execution

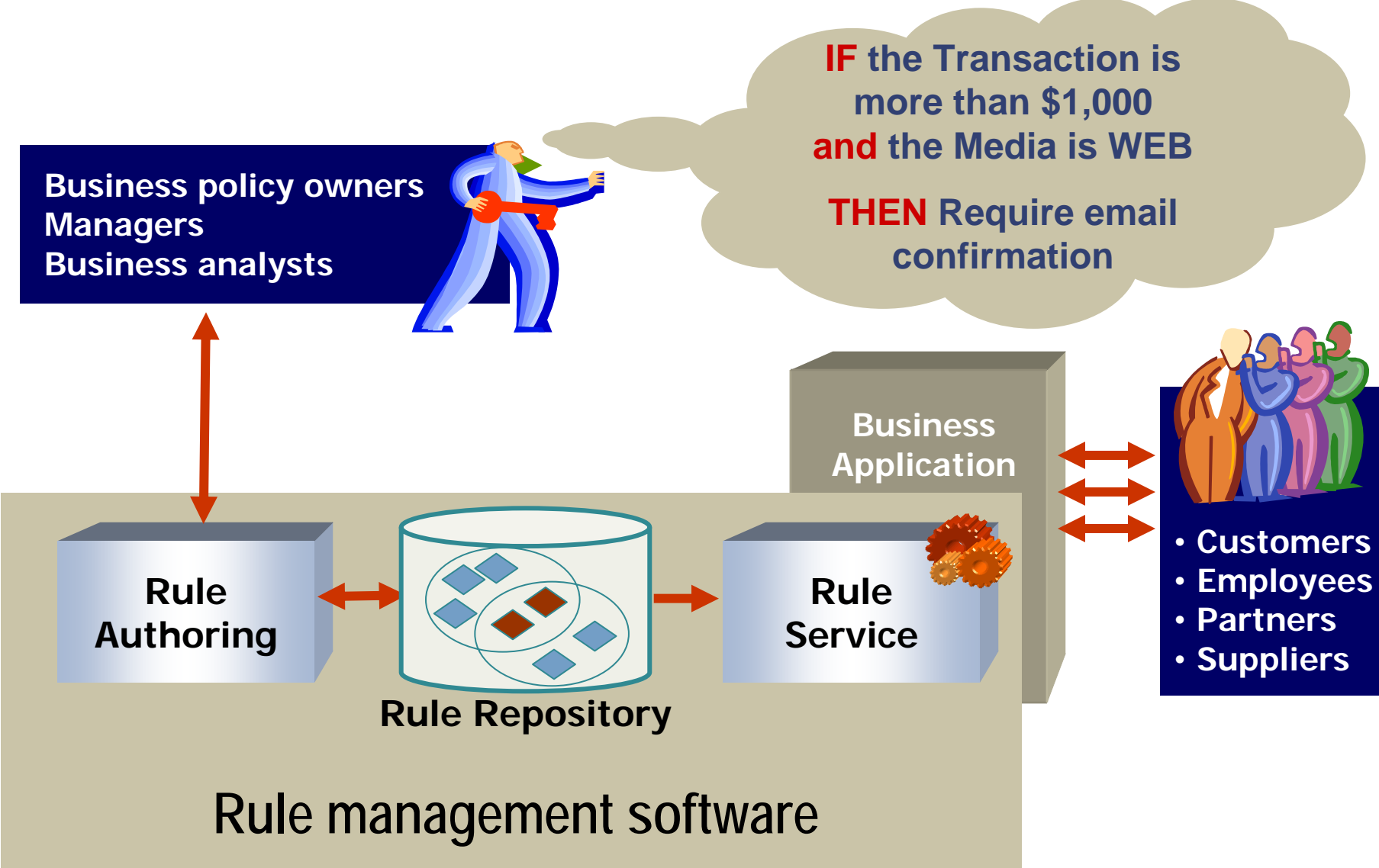


2000s Problem:
rule lifecycle management

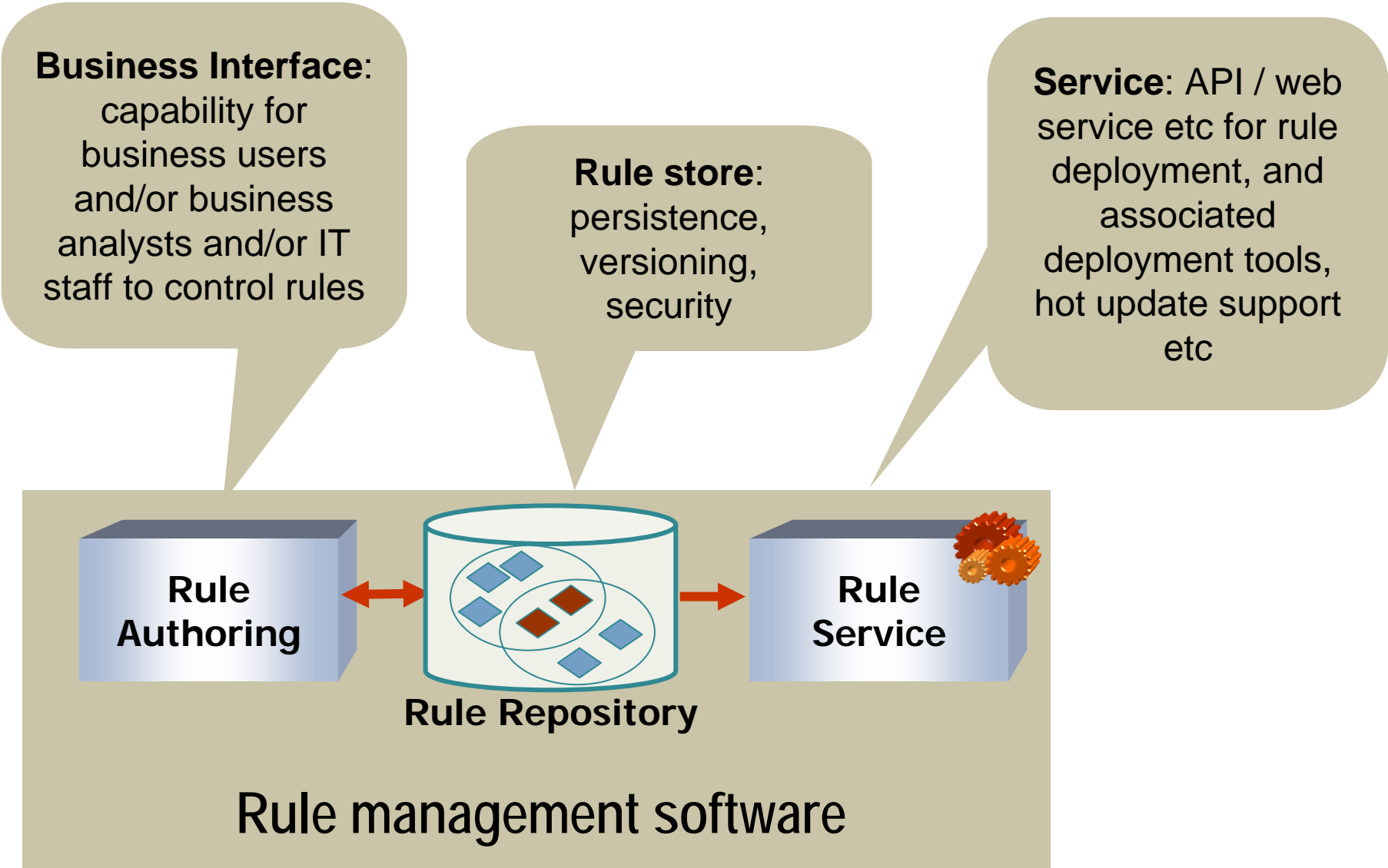
“Business Rule Management”

what it means for IT

How does it all fit into applications?



IT view on BRM implementations



Software Best Practices vs Rule Management?



1. Develop Software iteratively
2. Continuously verify software quality
3. Control changes to software
4. Manage Requirements
5. Use Component-based architectures
6. Visually model software

How does Rule Management fit?

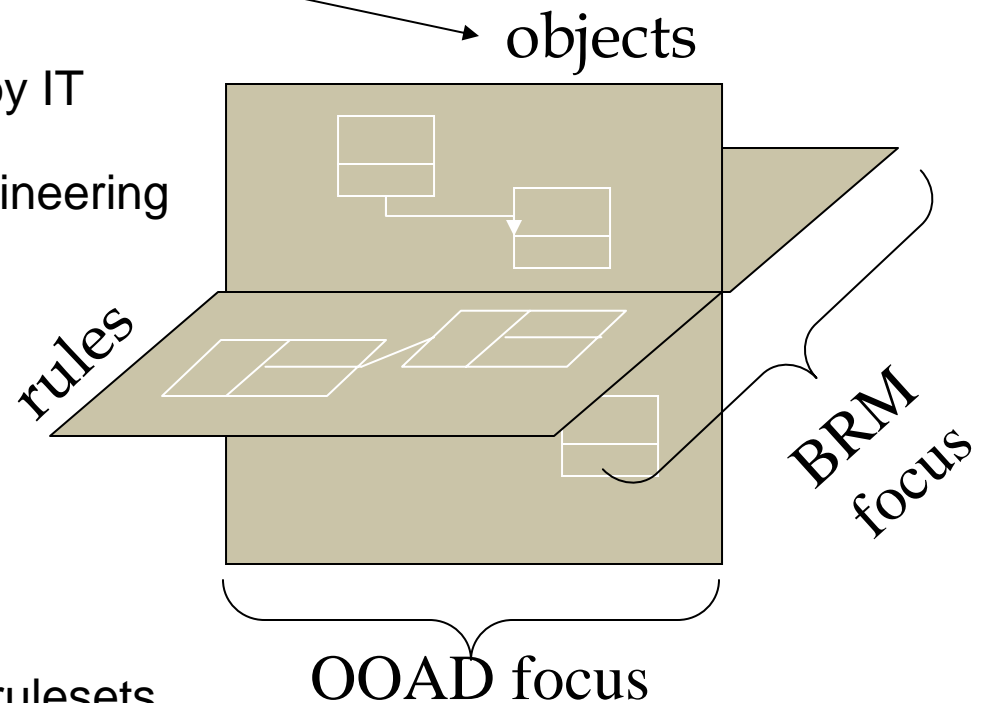
Crossover of OO and Rules

▶ OO Development

- Defines business object model and infrastructure code
- Modeled in UML, developed by IT
- Subject to many software engineering best practices

▶ Rule Management

- Declarative rules
- Defined in and outside of IT
- Organized into services, and rulesets
- Dependent on business object model



S/W Best Practices (1)

1. Develop Software iteratively

- Implication: break down s/w development into manageable & measurable “chunks”
- Declarative business rules are individual, testable units of decision making
- BRMS allows the incremental development / test of business rules, separate from the other s/w parts

2. Manage Requirements

3. Use Component-based architectures

4. Visually model software

5. Continuously verify software quality

6. Control changes to software

S/W Best Practices (2)

1. Develop Software iteratively
2. Continuously verify software quality
 - Software approval / walk-through / test cycles
 - Business rules in near-English are easier to check
 - BRMS encourages constant rule validation and verification cf RAD, XP
3. Control changes to software
4. Manage Requirements
5. Use Component-based architectures
6. Visually model software

S/W Best Practices (3)

1. Develop Software iteratively
2. Continuously verify software quality
3. Control changes to software
 - Software systems are “brittle” – small changes can break them
 - ... yet “change” is guaranteed and must be managed
 - Business rules as declarative logic statements are designed for change
 - BRMS include higher-level rule maintenance facilities (eg Rule Maintenance Applications, Business Languages ...)
4. Manage Requirements
5. Use Component-based architectures
6. Visually model software

S/W Best Practices (4)



1. Develop Software iteratively
2. Continuously verify software quality
3. Control changes to software
4. **Manage Requirements**
 - Requirements = use cases + descriptions, behavior
see “Use Cases – Requirements in Context”
 - Business view:
Requirements = business rules + system / process needs
 - BRMS provide a means of managing the rules in
“requirements” in the form of explicit business rules,
throughout the application lifecycle
5. Use Component-based architectures
6. Visually model software

S/W Best Practices (5)

1. Develop Software iteratively
2. Continuously verify software quality
3. Control changes to software
4. Manage Requirements
5. Use Component-based architectures
 - Divide-and-conquer approach to software development + re-use technical components off-the-shelf (eg a RDBMS for data management)
 - Separation of business rules from data, UI, middleware etc into its own component makes sense
 - BRMS include various rule execution services supporting different platforms and architectures
6. Visually model software

S/W Best Practices (6)

1. Develop Software iteratively
2. Continuously verify software quality
3. Control changes to software
4. Manage Requirements
5. Use Component-based architectures

6. Visually model software

- “A picture is worth a 1000 words” – visualization is necessary for good communications (eg UML class diagrams)
- Business rules and their interrelationships, cross-references to object models, and control flow should also be visualized
- BRMSs provides powerful UI tools for modeling and viewing rules and their interactions with data

S/W Best Practices vs BRE and BRM



- ▶ Summary: a fit!

POLL

▶ HOW MANY:

▶ Have software systems that require updates to business logic

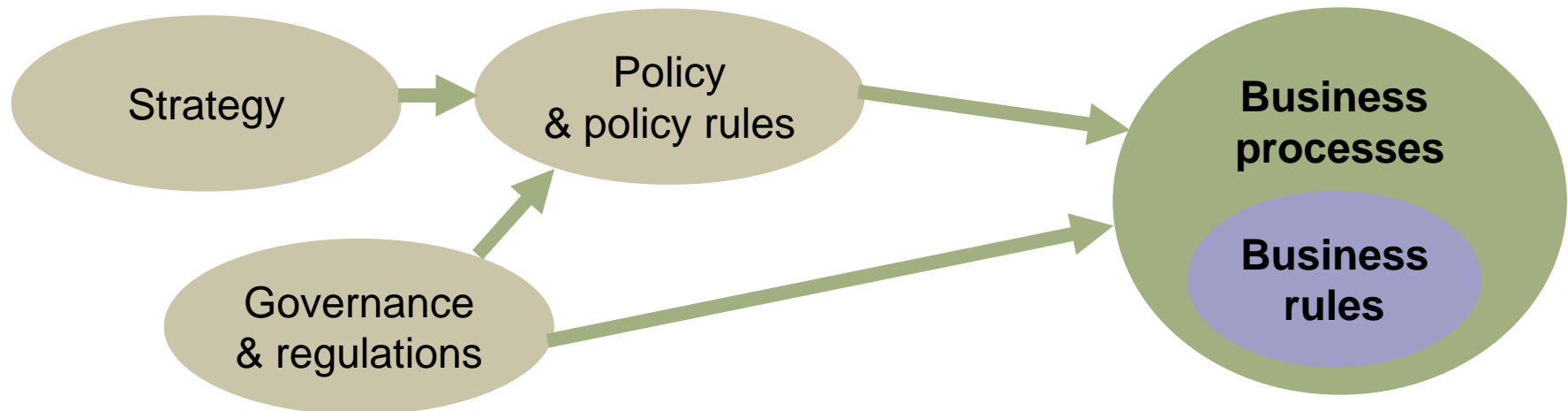
1. Never
2. Once every few years
3. A few times a year
4. A few times a month
5. Every week or day!

Business “Rule Management” vs “Process Management”

what are the differences?

Processes vs Rules

- ▶ Processes are higher level than rules
 - ▶ Rules can be used to implement processes
 - ▶ Rules are usually exposed as services: processes consume services



See also Business Motivation Model
at <http://www.businessrulesgroup.org/brgactv.htm#wp2>

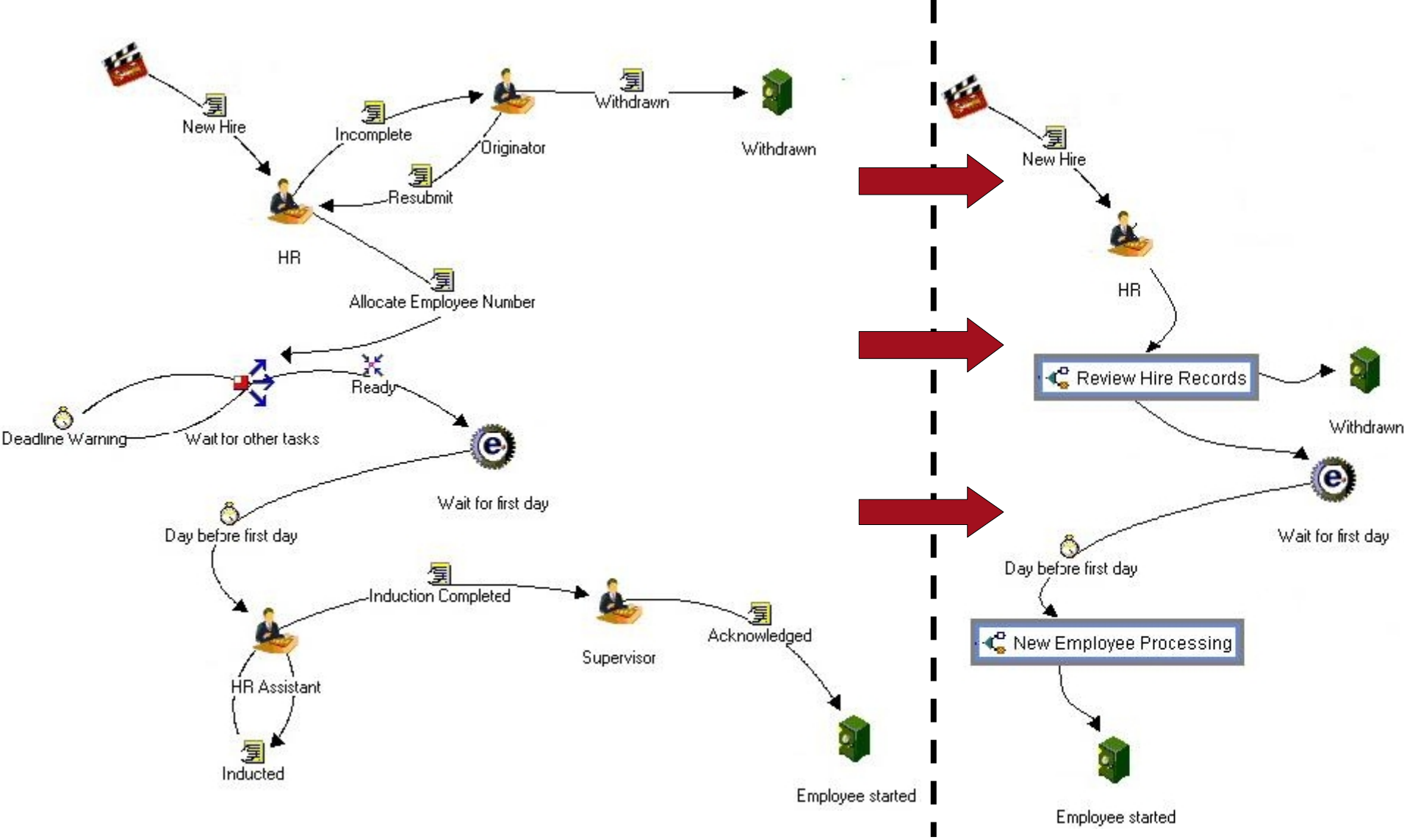
BPM vs BRM

- ▶ BPM is a w-i-d-e term
 - ▶ Business process modeling and simulation
 - ▶ Business process orchestration and flows
 - ▶ Business process automation and manual workflow
 - ▶ Business Performance Monitoring ☺

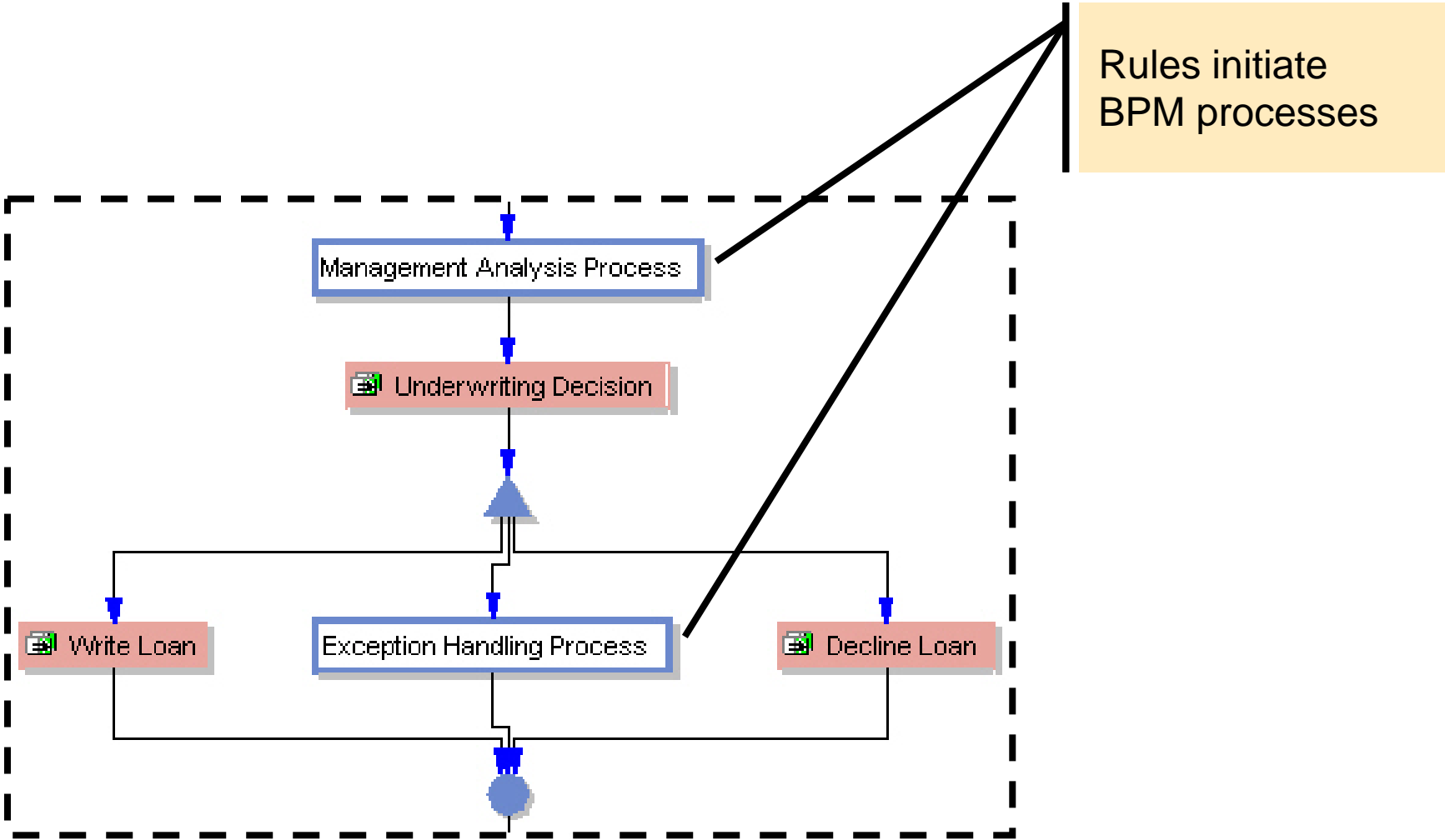
- ▶ Focus on Process definition and execution aspects

- ▶ BRM is much the same

Rules Automate and Reduce BPM Tasks



BRMs Can Include BPM Processes



Benefits?



<<In a study, undertaken by independent research company Strategic Focus, of Malpitas, CA, it took 38% less time to build, deploy and test applications with a combined process and business rules environment than it did with a modern Java development environment. Further, it then took 58% less time to change the completed application.>>

Business Rules are from Mars & Processes from Venus by Derek Miers

Summary



- ▶ Business rules, rule engines and rules management:
 - ▶ Good fit with SOA and BPM
 - ▶ Good fit with IT
 - ▶ Plenty of tool support (from open source to full-house)
 - ▶ Increasing interest from Big 3 (MS, IBM, Oracle)

Thank you

References:

www.brcommunity.com

www.businessrulesgroup.org

Authors on this topic:

www.brsolutions.com & www.kpiusa.com

Events:

www.businessrulesforum.com & <http://www.eurobizrules.org/>

Your presenter: paulvincent@fairisaac.com of www.fairisaac.com/rules
(for the Blaze Advisor business rule management suite)

Q&A

ADVANCED WARNING:

Proposal to set up a specialist BCS group on Business Rules:
contact myself to receive updates on this...