Architecting For Change

architecting for change

“our business is constantly changing to meet new demands of the marketplace”

“we need faster time to market to remain competitive”

“our plan is to engage heavily in mergers and acquisitions”

architecting for change

change

architecting for change

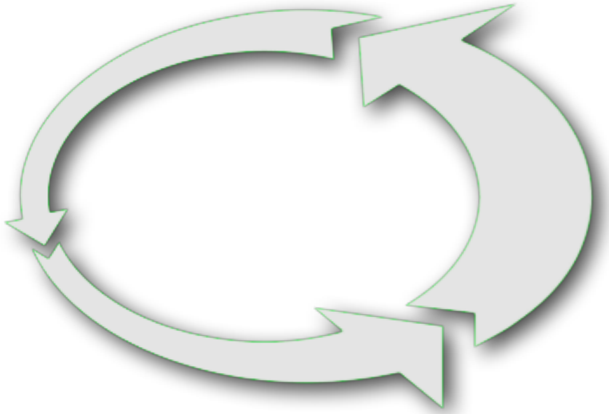
traditional methods of architecture are not sufficient to meet the ever-changing demands of the marketplace

architecting for change

business is in a constant state of change

increased competition

regulatory changes



mergers

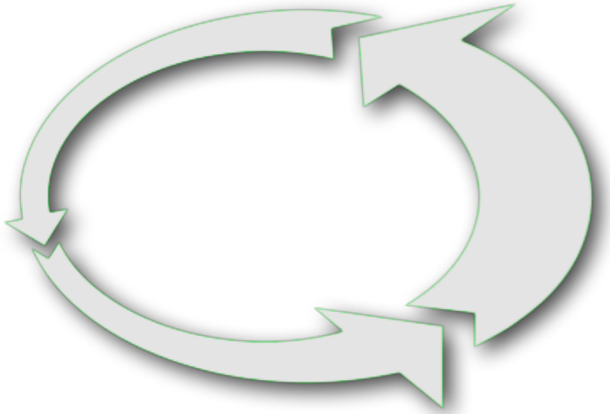
growth

acquisitions

architecting for change

technology is in a constant state of change

platforms



products

languages

patterns

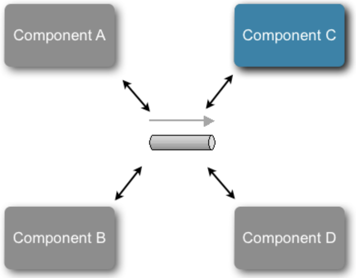
frameworks

architecting for change

architecture agility

the ability to respond *quickly* to a constantly changing environment

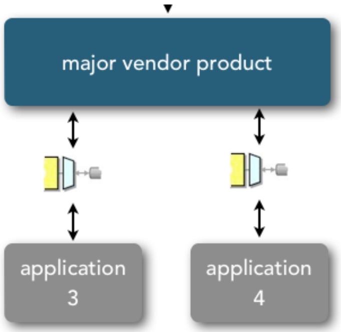
techniques for change



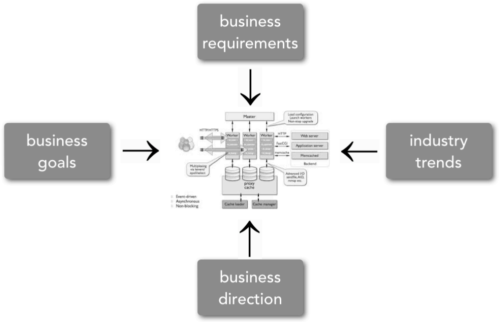
reduce dependencies

SWIFT
SOAP
FIX PROTOCOL
INDUSTRY-DRIVEN MESSAGING STANDARD℠
<?xml?>
FpML
Financial products Markup Language

leverage standards

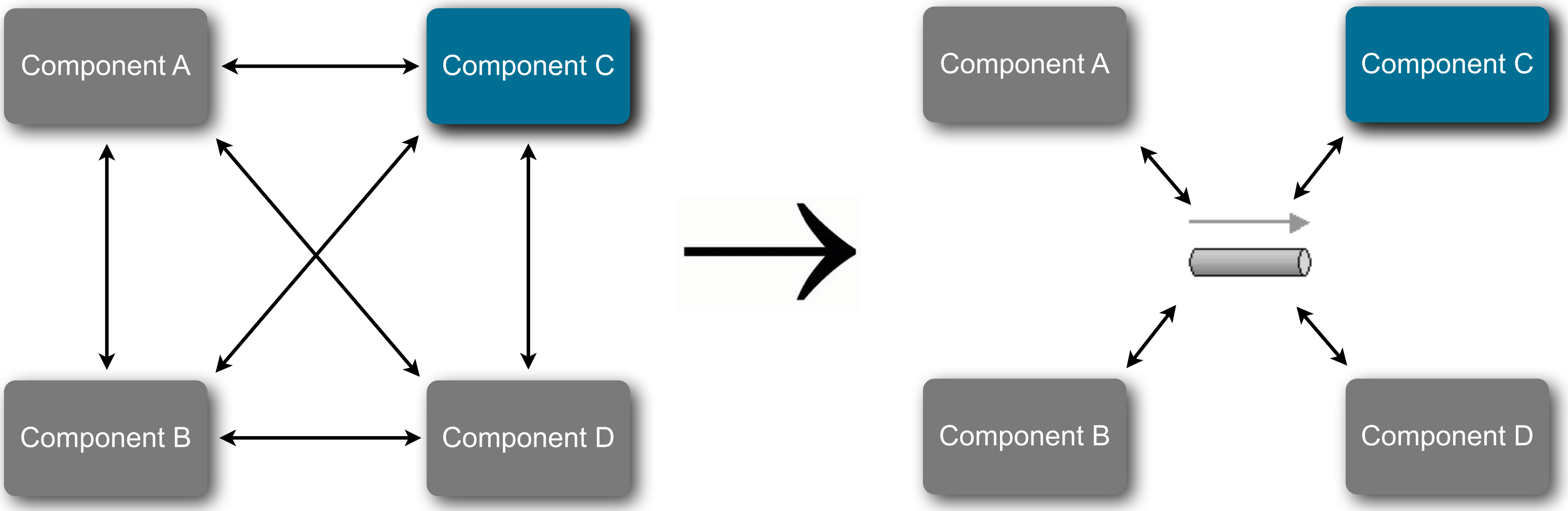


create product-agnostic architectures



create domain-specific architectures

reduce dependencies

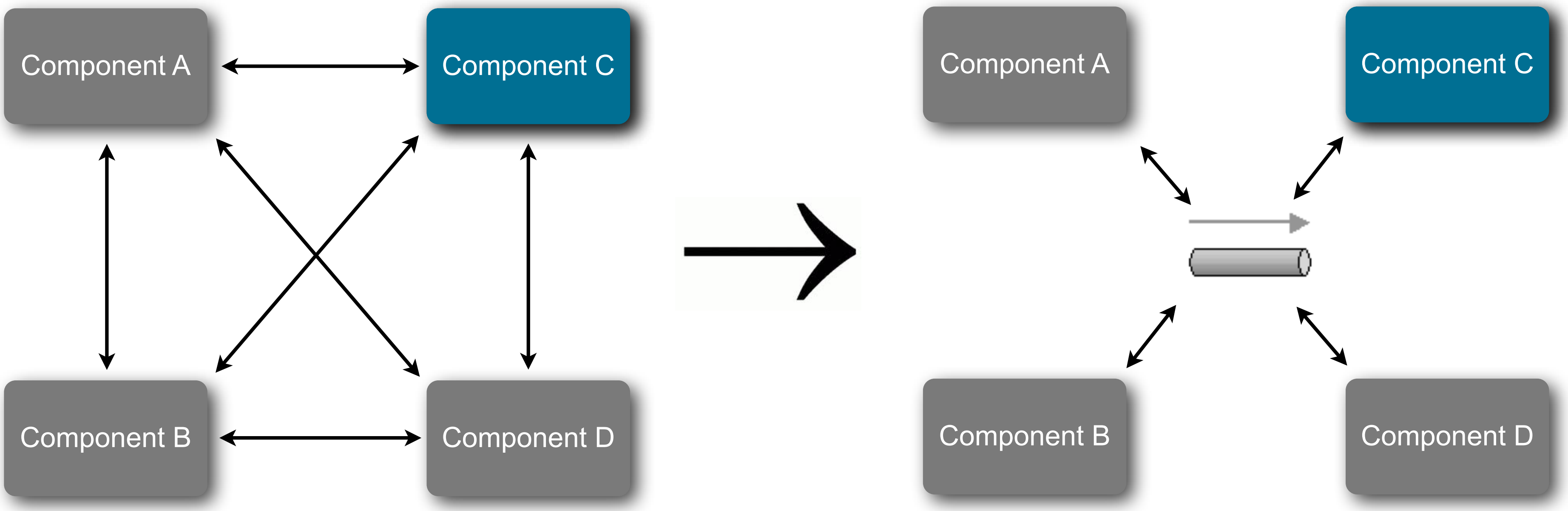


less coordination required (release schedules)

components can evolve independently

faster deploy [illegible]

reduce dependencies



messaging service bus adapters

architecture patterns

leverage standards

industry standards

SWIFT

SOAP

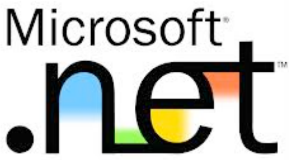
FIX PROTOCOL
INDUSTRY-DRIVEN MESSAGING STANDARD℠

<?xml?>

FpML
Financial products Markup Language

leverage standards

corporate standards



leverage standards

de-facto standards

HIBERNATE

Jakarta Struts

Apache Tomcat

spring

leverage standards

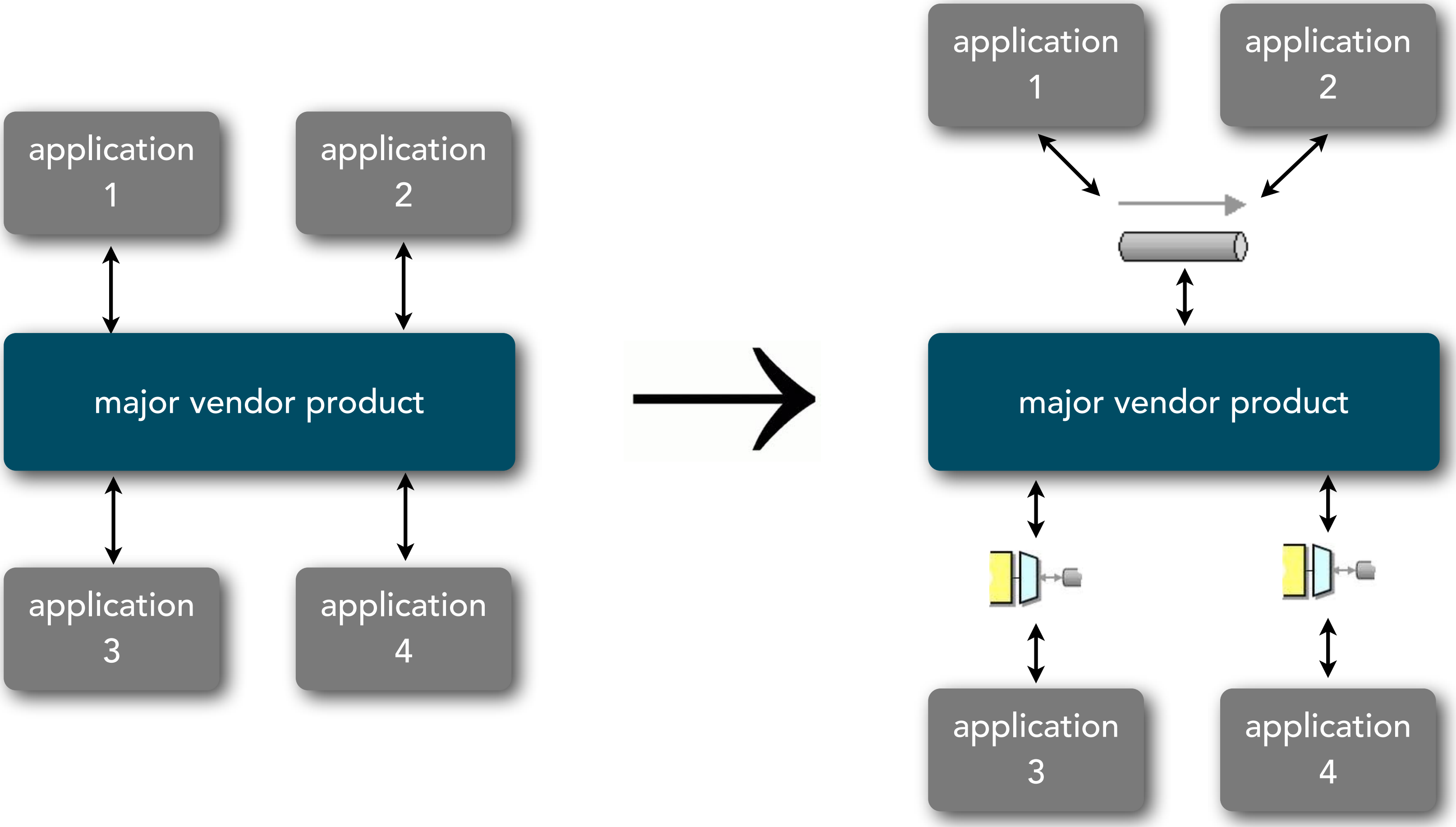
standards may not always be your first choice, but they significantly help in reducing the effort for change

larger resource pool

better integration with other systems

product-agnostic architecture

isolate products to avoid vendor lock-in

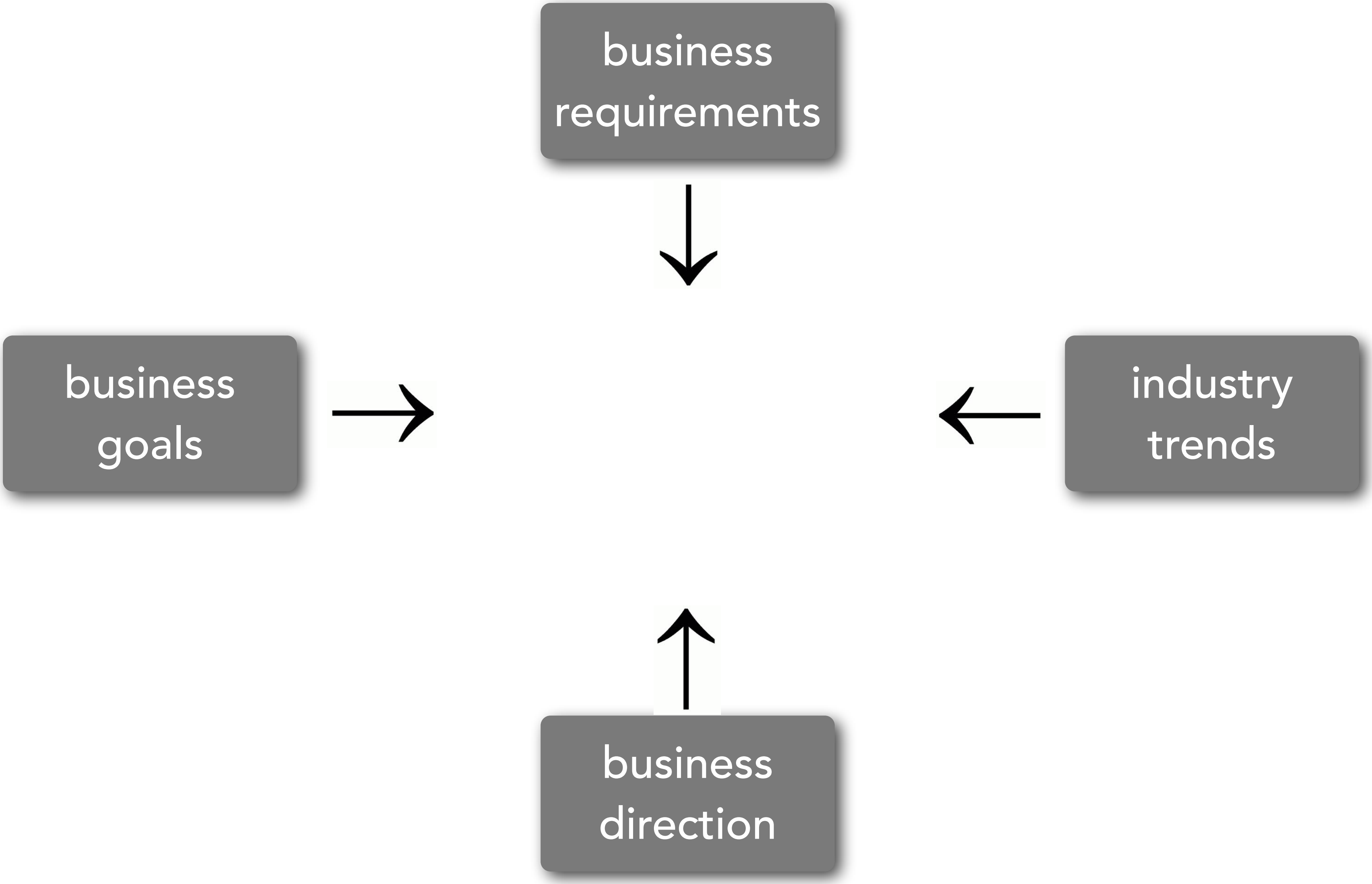


domain-specific architecture

generic architectures are difficult to change because they are too broad and take into account scenarios that aren't used

limit the scope of the architecture by taking into account drivers, requirements, business direction, and industry trends

domain-specific architecture



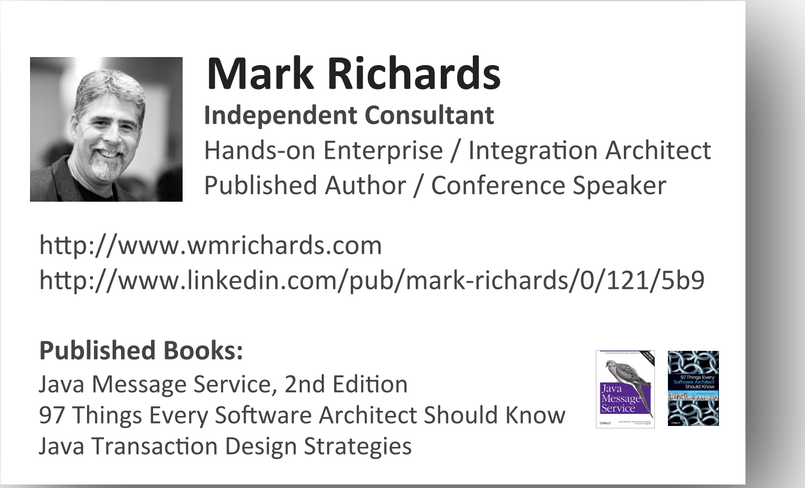
business requirements

business goals

industry trends

business direction

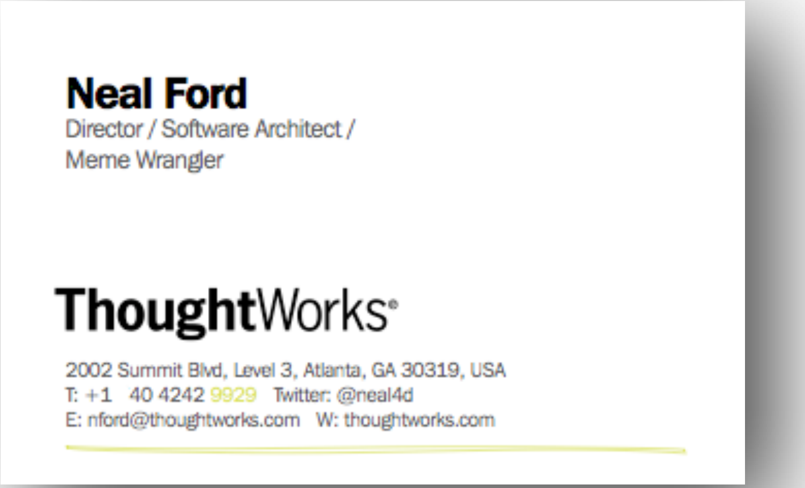
?’s



Mark Richards
Independent Consultant
Hands-on Enterprise / Integration Architect
Published Author / Conference Speaker

http://www.wmrichards.com
http://www.linkedin.com/pub/mark-richards/0/121/5b9

Published Books:
Java Message Service, 2nd Edition
97 Things Every Software Architect Should Know
Java Transaction Design Strategies



Neal Ford
Director / Software Architect /
Meme Wrangler

ThoughtWorks®

2002 Summit Blvd, Level 3, Atlanta, GA 30319, USA
T: +1 40 4242 9929 Twitter: @neal4d
E: nford@thoughtworks.com W: thoughtworks.com