



WEB AGE
solutions inc.

A Composable Service is a Good Service

Kyle Gabhart

Director of SOA Solutions for Web Age Solutions



Agenda

- ❑ **Introduction**
- ❑ Adventures in Service Orientation
- ❑ Profile Of A Composable Service
- ❑ Composable Services Are Good
- ❑ Governing Service Design
- ❑ Review

Shameless Plug

□ Who is Kyle Gabhart?

- ❖ Technology strategist and enterprise architect with a broad range of relevant experience
- ❖ Currently working with several Fortune 500's on their SOA strategies, including CalPERS and PFG as well as several state and federal agencies in the U.S. and Canada
- ❖ Author of nearly 100 articles, white papers, books, and training programs



- ❖ Open source contributor, consultant, architect and strategist on SOA and Web services since 2001
- ❖ Author of *soamatters.com*
- ❖ SOA Solutions Director for Web Age Solutions, a leading education and mentoring firm



Agenda

- Introduction
- **Adventures in Service Orientation**
- Profile Of A Composable Service
- Composable Services Are Good
- Governing Service Design
- Review

Adventures in Service Orientation



See Dick.

See Dick Code.

See Dick's Code Run.

See Dick Wrap His Code With A Web Service.



See Jane.

See Jane Code.

See Jane's Code Consume Dick's Service.

See Jane Get Frustrated Because Dick's Service Requires That Certain Operations Be Called In A Particular Order And Other Operations Should Only Be Called Within a Business Process With Certain Business Logic Parameters Set By The Process.

Jane Is Mad At Dick.

Dick Has Built A Service That Is Difficult To Use.

Don't Be A Dick.

More Adventures in Service Orientation



See Dick.

See Dick Write Another Service.

See Dick's Service Require Several Operations Be Called Within A Distributed Transaction Context In Order To Ensure That Enterprise Data Is Left In A Consistent State.



See Jane.

See Jane Consume Another Service From Dick.

See Jane's Code Corrupt Data And Violate Several Business Rules Within The Company Supply Chain.

See Jane Study The Service Interface And Service Contract To Verify That She Did Everything She Could To Avoid This.

Jane Is Very Mad At Dick.

Jane Will Ambush Dick At The Next Team Meeting And Make Him Look Foolish.

Run Dick, Run.



Agenda

- ❑ Introduction
- ❑ Adventures in Service Orientation
- ❑ **Profile Of A Composable Service**
- ❑ Composable Services Are Good
- ❑ Governing Service Design
- ❑ Review

Defining Terms

- Let's define some terms...

- According to Webster's Dictionary...
 - ❖ The term 'geek' derives from the German word for fool and can refer to:
 1. *a carnival wild man*
 2. *an expert or enthusiast in a technological field*
 3. *someone that spends two perfectly good days in Manhattan attending a conference on SOA*

 - ❖ But that's not important right now...



Defining *Relevant* Terms

❑ *Composable Service*

- ❖ A service that can readily be included in an aggregate service or business process without special considerations

❑ *Service Design*

- ❖ Encompasses the service granularity, messaging paradigm, and data model
- ❖ Typically represented by the service interface

❑ *Service Implementation*

- ❖ Encompasses the underlying technology implementation and platform scaling for a service
- ❖ Represented by the service code with quality of service parameters spelled out in policy files

Service design (not implementation) determines composability



What Makes A Service Composable?

- Composable services make no assumption about the context under which they will be called
 - ❖ Service operations can be called in any order desired
 - ❖ Once service operations complete, enterprise data is always left in a valid state
 - ❖ Service operations do not care whether or not they are called from within a transaction context
 - ❖ Consumers are free to use a service operation to meet whatever business need they have (*rather than the business need that the service developer originally had in mind*)

- Composable services are designed to fulfill their service interfaces and nothing more
 - ❖ Service interaction is dictated by service interfaces
 - ❖ Providers and consumers are decoupled by interfaces

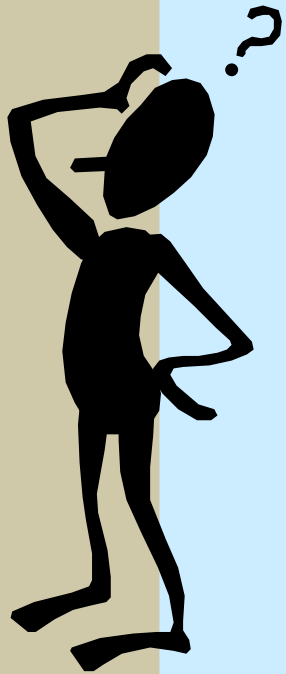


Agenda

- ❑ Introduction
- ❑ Adventures in Service Orientation
- ❑ Profile Of A Composable Service
- ❑ **Composable Services Are Good**
- ❑ Governing Service Design
- ❑ Review

Stories From The Front Line

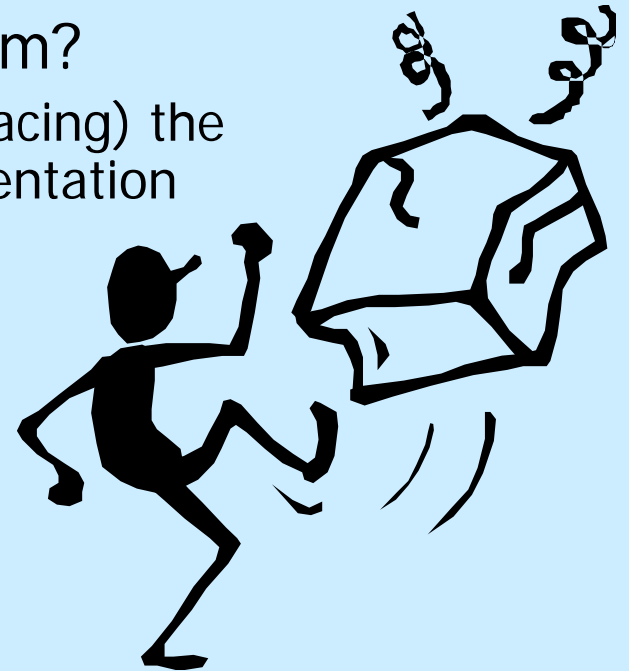
- ❑ We want for service providers and service consumers to have as little stress as possible
- ❑ Consider the following real-life scenarios from a recent client engagement
 - ❖ “We aren’t seeing an ROI because we spend so much time making sure that service operation calls do not leave the business in an acceptable state. State is verified before and after each call.”
 - ❖ Another person described needing to invoke a particular service operation that calls other operations and was concerned about those other operations being called independently.
 - ❖ A third person wanted to know what the best practice was for indicating that certain services should ONLY be called as part of a business process.



Non-composable Services Are Stressful

- Do you see a theme here?
 - ❖ All three scenarios depict services that do not have the right degree of encapsulation
 - ❖ Services designed in this fashion result in tightly-coupled, monolithic applications that are high maintenance and do not look much different from traditional application silos

- What is the heart of the problem?
 - ❖ A lack of understanding (or embracing) the fundamental tenets of service orientation
 - ❖ Service oriented solutions require new ways of thinking and new design strategies



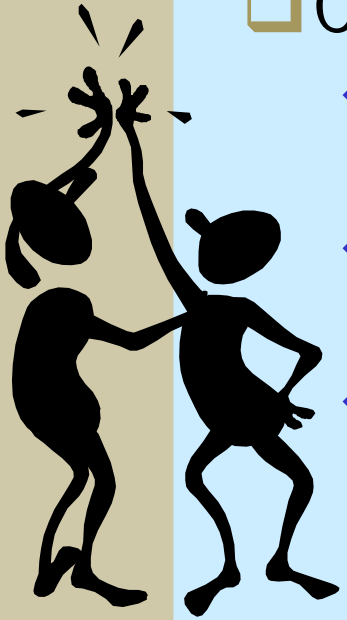
Composable Services Are Good

- Services should be discreet and loosely-coupled
 - ❖ There should be no 'special' logic that must be applied
 - ❖ The service interface, associated metadata, and service implementation should be entirely self-sufficient
 - ❖ Composable services empower consumers to develop a unique solution, rather than working with pre-configured processing blocks that are inflexible

- IBM's definition for Web Services back in the dark ages (circa 2004) is still a good guide
 - ❖ "...a new breed of Web-based application that are ***self-contained, self-describing, modular*** applications..."

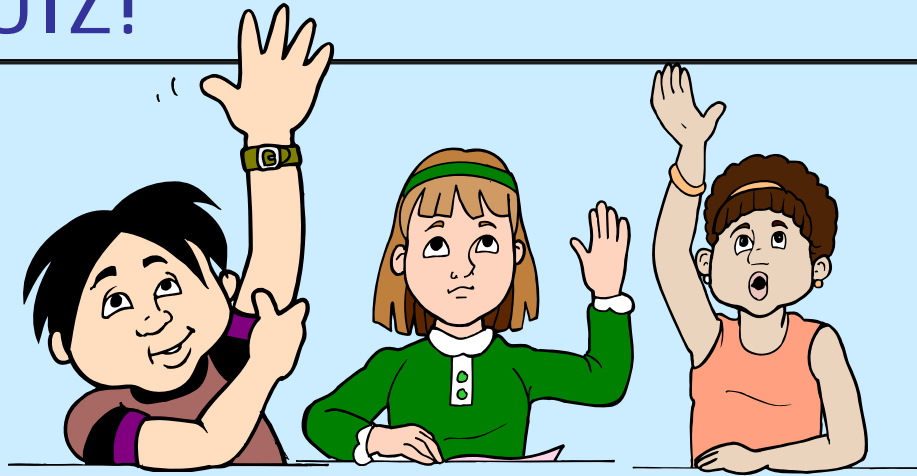


Everybody Wins

- 
- ❑ Composable services are good for consumers
 - ❖ Service operations can be used independently, called in sequence, or never called at all
 - ❖ Services can be included or excluded from a business process as needed
 - ❖ Solutions can be composed of best-of-breed services and business processes

 - ❑ Composable services are good for providers
 - ❖ Publish service interfaces, along with supporting contracts, SLAs, and policies
 - ❖ Rely upon the service infrastructure and runtime governance to protect the enterprise

POP QUIZ!



- Is a service designed to be composable or implemented in a composable way?
- What two primary qualities do we associate with composable services?
- What is so great about composable services?



Agenda

- ❑ Introduction
- ❑ Adventures in Service Orientation
- ❑ Profile Of A Composable Service
- ❑ Composable Services Are Good
- ❑ **Governing Service Design**
- ❑ Review

Beginning With the End in Mind

- ❑ Good architecture is not accidental
 - ❖ In traditional buildings, architecture is not an after thought that occurs sometime after the foundation has been laid, it is designed from the beginning
 - ❖ The architecture provides the framework off of which everything else hangs
- ❑ Good architecture is business-centric
 - ❖ Architecture should be sufficiently robust to support the future direction of the organization
 - ❖ Architecture should enable and protect the enterprise business drivers





Governing Service Design

- Governing the design of composable services involves several steps
 - ❖ Evaluate business drivers
 - ❖ Determine the importance of composability
 - ❖ Put metrics in place to gauge effective service design
 - ❖ Govern service granularity and atomicity
 - ❖ Promote full lifecycle service design that emphasizes contractual service design



Evaluate Business Drivers

- ❑ Before you govern service design, you need to know what your goals are for service orientation
 - ❖ Are you aiming for more agile business processes?
 - ❖ Is service orientation being adopted to reduce systems integration and maintenance costs?
 - ❖ Is service and process reuse important for you?
 - ❖ Are you trying to break down silos and facilitate alignment and interoperability across silos?

- ❑ Next you need to identify quantifiable metrics associated with your business drivers
 - ❖ Service reuse factors
 - ❖ Total Cost of Ownership (TCO) for systems or processes
 - ❖ Time to market
 - ❖ Quality of Service (QoS)
 - ❖ And etc.



Determine Importance of Composability

- How important is it for your services to be composable?
 - ❖ Composable services enable agility
 - ❖ Composable services help to keep integration costs low
 - ❖ Composable services are easier to reuse

- Then you'll need to determine how important this is for you
 - ❖ Do you need strict governance to produce highly composable services?
 - ❖ Will looser design guidelines be sufficient to provide a moderate degree of composability?



Designing Composable Services

- ❑ There are several factors to pay close attention to when designing composable services
 - ❖ Service granularity
 - ❖ Service atomicity
 - ❖ Contract-driven services

- ❑ You will want to put guidelines and metrics in around these key topics

Service Granularity

- *Service granularity* – a measure of how broad the interaction between service consumer and service provider must be to meet the requirements
 - ❖ fine-grained services represent relatively small amounts of functionality and exchange small data sets
 - ❖ coarse-grained services employ higher levels of abstraction, encapsulating large sets of functionality within a single interaction

Example:

A *fine-grained* Weather service might provide you with **select data** such as temperature or wind sheer or barometric pressure

A *coarse-grained* Weather service might give you **all weather related information** for a particular area

Service Atomicity

- *Service atomicity* – an indication of whether a service can fulfill requests directly or whether it must leverage other services
 - ❖ atomic services are self-sufficient, depending only upon the local service implementation to process the service request
 - ❖ composite services a collection of services that collectively fulfill the consumer's request

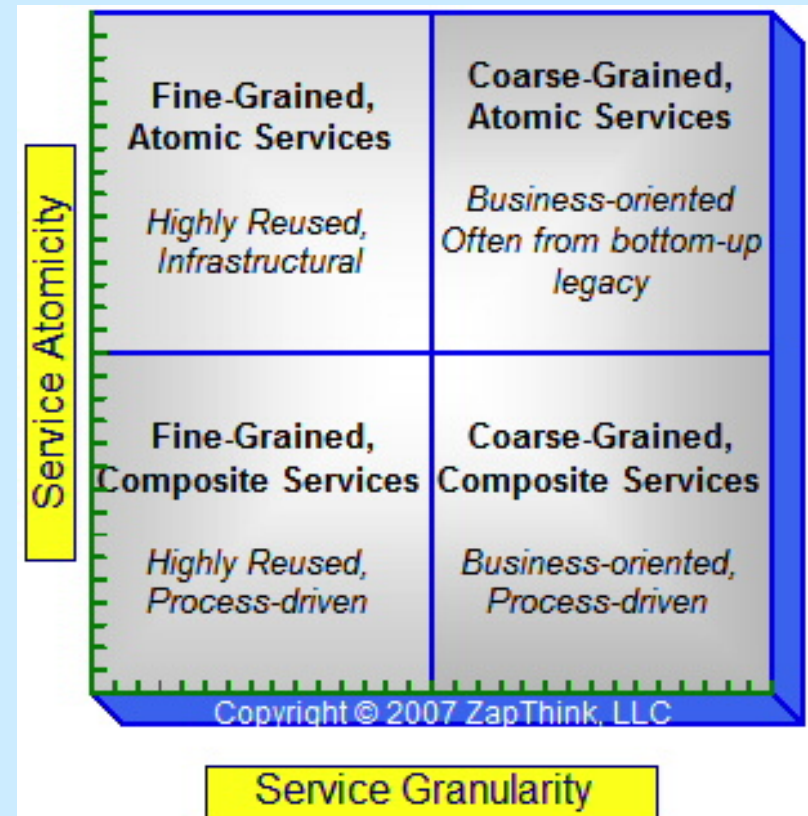
Example:

An *atomic* Weather service might provide you with data from a **single source** such as NOAA, your airport, or local media outlet

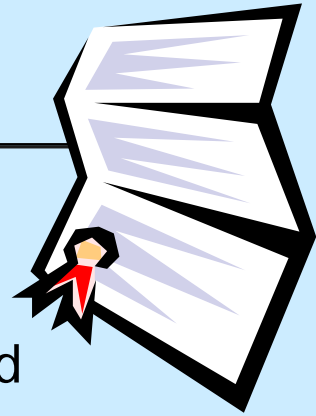
A *composite* Weather service might give you weather information **aggregated from multiple sources**

Service Granularity Matrix

- ❑ ZapThink has proposed the following matrix for understanding service granularity and atomicity



Contract-driven Services



□ Design-time

- ❖ Verify and review service contract and SLA's
- ❖ Ensure that applicable policies have been identified
- ❖ Ensure that design artifacts are consistent with recommended guidelines and best practices

□ Change-time

- ❖ Verify that service contract, interface, and policies are still in sync
- ❖ Define and apply service management strategy
- ❖ Deploy service when all service management concerns (versioning, registering, etc.) have been addressed

□ Run-time

- ❖ Automatic policy enforcement
- ❖ Periodic production environment auditing, especially against SLAs
- ❖ Care and feed production environment (respond to events, monitor process and service health, etc.)



Agenda

- ❑ Introduction
- ❑ Adventures in Service Orientation
- ❑ Profile Of A Composable Service
- ❑ Composable Services Are Good
- ❑ Governing Service Design
- ❑ **Review**

Wrap-up

- ❑ Questions, comments, thoughts or concerns?



- ❑ *To explore this subject further, stop by my blog, SOA Matters – www.soamatters.com*