SIEMENS

SOA in HIS Development

April 17, 2008

Dave Torok
Alex de Jong
Pavel Kilian

SIEMENS

Outline

- Reasoning for SOA and Business Value
- Introducing a Clinical Case Study
- SOA Design Tradeoffs In Events and Services Design
- Evolution and Trends

SIEMENS

Our View on SOA

- A Service-Oriented Architecture (SOA) is a software architecture that is based on key concepts of application front-ends, services, a service repository, and a service bus
  - A **service** is functionality from a software component that is exposed through a set of **contracts or interfaces** providing a stated behavior through the use of business logic and/or data access
  - A **service repository** provides facilities to discover services and acquire information necessary to use the services
  - An **application front-end** initiates and controls all activity of the enterprise system
  - A **service bus** connects all participants of an SOA
- The concept has been around for a long time
- It is a principle around which our applications are designed
- It is infrastructure which applications utilize to assist their achievement of both functional and non-functional requirements
- Includes not just service calls, but also messages and events

*Definitions from: Enterprise SOA: Service-Oriented Architecture Best Practices; Dirk Krafzig, Karl Banke, and Dirk Slama; Prentice Hall, 2005.

SIEMENS

Reasoning for SOA – Business Value

Customer Perspective

- Customers don't desire, choose or buy architecture, they want differentiating features that clearly:
  - Help improve patient clinical outcomes, help save lives
  - While Improving Operational efficiency and profitability/viability
- Rich Features implemented as "Micro Workflows"
  - One person accomplishing a single task efficiently in one system at one time
  - Don't require SOA
- "Macro Workflow Orchestration" – enabled by SOA
  - Customer Configurable
  - Multiple products and systems that don't normally interoperate
  - Cross multiple persons and domains
  - Executing an objective over time – long running processes
  - … to improve Patient Outcomes / Operational Efficiency

SIEMENS

Reasoning for SOA – Business Value

Software Provider Perspective

- Modularity
- Speed – reuse of services
- Speed – isolated dependencies and loose coupling
- Supportability – reuse: less code, fewer bugs. Isolated dependencies, faster bug isolation/resolution…
- Cost – standard and patterned approach to infrastructure

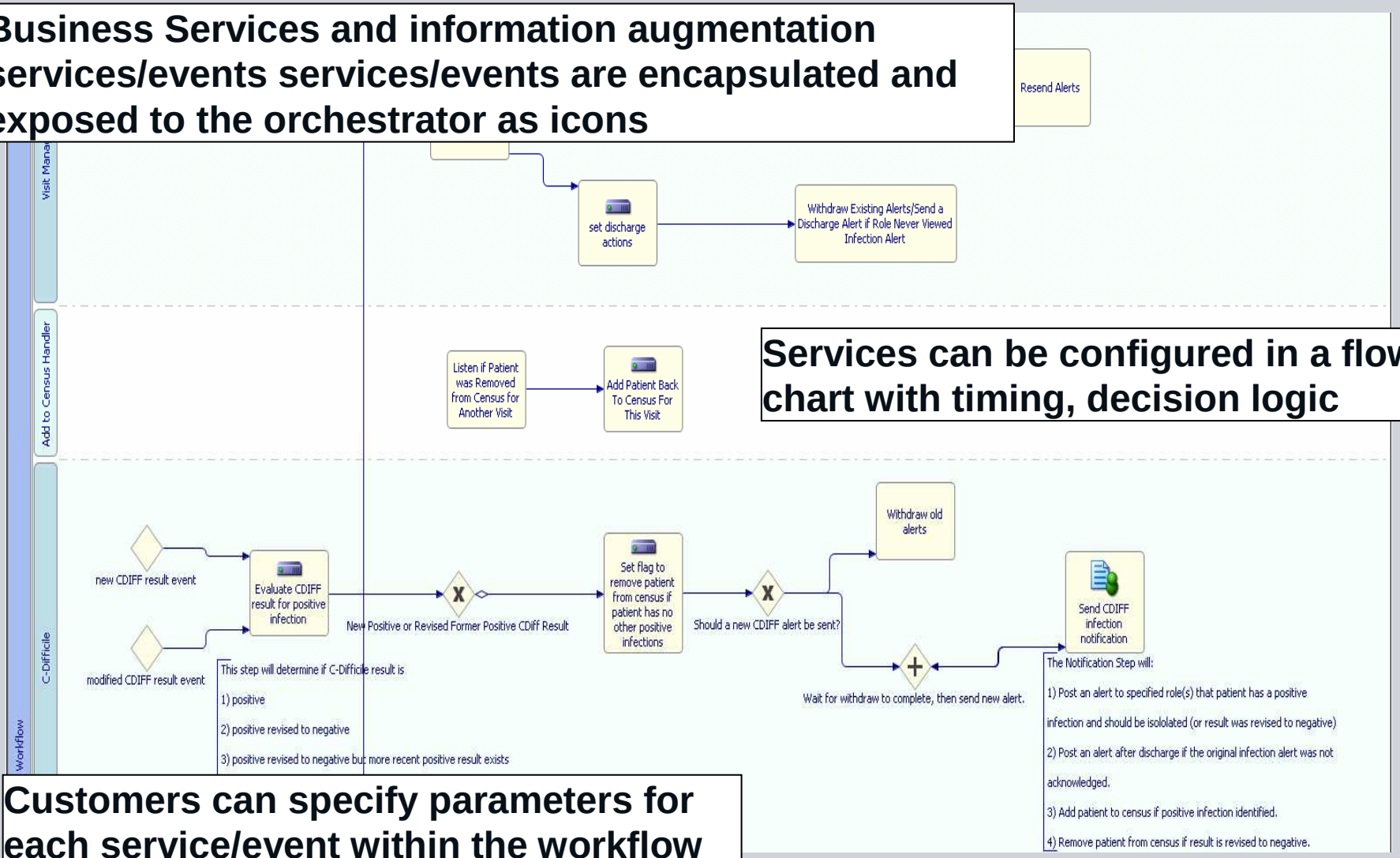
SIEMENS

Seeking those benefits – Historical View

- Technical Infrastructure
  - There were not comprehensive SOA technical standards 5+ years ago
  - Distributed architectures: CORBA, Java RMI (homogeneous), message brokering
  - Proprietary RPC protocols; widespread adoption in Siemens products
  - Internal SOA (within a product)
- Logical Infrastructure
  - Services exposed tactically (bottom up) to support specific workflows (based on customer needs) vs. systemically by model / business service creation
  - Service definitions for workflow orchestration vs. code modularity may not correlate – less reusability
  - Not a deliberate SOA service library, but a library of workflow building blocks
  - Often used internal IDs (instead of external)
- Operational Infrastructure
  - Inventory, meta data (for discovery), contracts, etc. not exposed

Infection Control Workflow

Business Services and information augmentation services/events services/events are encapsulated and exposed to the orchestrator as icons



Services can be configured in a flow chart with timing, decision logic

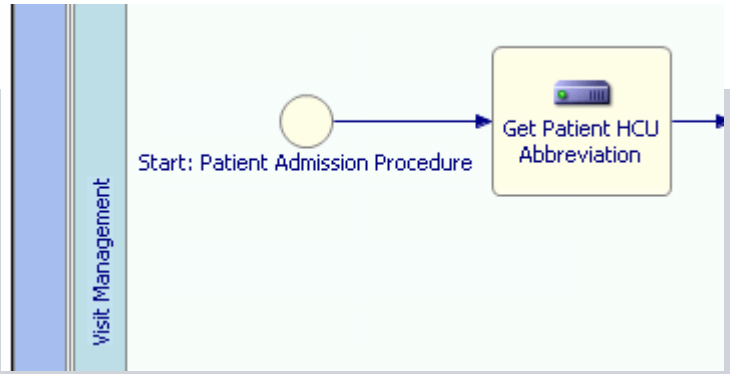
Customers can specify parameters for each service/event within the workflow

SIEMENS

Infection Control Workflow

- The Infection Control workflow starts upon patient presentation to the facility. It is used for all admission types.
- It seeks to identify patients who have tested positive in the past or who do text positive in the current stay for specific infectious indicators
  - The workflow monitors three parameters: C-Difficile, MRSA, and VRE
  - The workflow allows a customer to modify (add, delete, change) the indicators
- Upon identification of a patient meeting the infectious disease protocol
  - The patient is added to staff census list
  - Staff member(s) are notified
  - A patient location change may be suggested. Special housekeeping instructions may be suggested
  - New cases may suggest medication treatment orders
  - Escalation clocks are started (listening for a patient transfer execution)
- The workflow (or linked workflows) handles various “What If” scenarios
  - What if the patient is discharged? What if the patient is merged? What if the result is revised to negative?

Workflow Triggering Events



- Patient “Admission”
  - Workflow applies to all encounter types (in patient, outpatient, emergency, etc.)
  - Is more accurately (less precisely, more generally), new patient encounter
- Multiple Sources of Events
  - Patient registered via Web UI (clinical system event)
  - Patient registered in another system (Master Patient event) Master Patient sends HL7 A01 or A04
- Granularity
  - “Admit Patient” is an encapsulated business service that, when designed for application integration is rich in data, worries about encounter types, validating medical responsibility, validating patient location availability, etc.
- Richness
  - For the purpose of initiating *this* workflow, it’s a thin event of just a state change – simply that there is a current intersection between patient and healthcare organization

SIEMENS

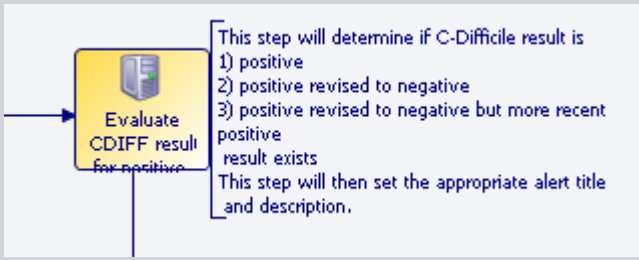
Design Considerations from Triggering Events

- Multiple Sources of Events: Used "Slave" ADT events emitted under our control
  - Listen to all admission sources or intercept and normalize to a "new patient" event?
  - Advocate SOA-Standardized event from the Master ADT system as source of truth
- Granularity: Used finer-grained events than "Admit Patient", at the native level of the encounter management business process
  - No additional burden on primary domain to abstract. Consumer responsibility to subscribe to all events that aggregate to "Admit Patient"
  - Consumer needs foundational knowledge of other domains' processes
  - Fine-grained events allow better discrimination by the consumer for items of interest. Trade-off is frequency of events.
  - Advocate SOA Infrastructure (Event Routing, Transformation, Aggregation) to eliminate burden from Provider or from Consumer applications.
- Richness: Used events just rich enough to describe what happened
  - Middle ground between "something happened" and sending all patient information
  - Advocate SOA events to be data-thin capturing just the information that defines the event process and allows consumers to make a decision whether they need to care.
  - Data thin eases the backward compatibility and migration burdens

Service and Event Analysis to identify a patient with infectious disease

SIEMENS

- Service call to determine history of disease: “Get results for patient” business service to determine if there has been a history of infectious disease
- Event listening to determine new test results that would indicate new instance of disease: “New Result” event listening
- Service Request, Event Subscription, and Evaluation Logic Parameters: Patient ID, Date Range, Disease(s) of interest, Result value for diseases



SIEMENS

Service and Event Design Considerations

- Location of Domain Knowledge
  - Which domain has the disease information? Results? Patient Problem List?
  - Centralized, "Source of Truth" with runtime services
  - Distributed, use source data result events and services
    - Issues: Volume, currency of data, is data universally available
    - Past results may have associated Problems, but during an encounter results are most current
  - Chose Results domain due to inconsistent use of patient problem list.
    - Result data is source data for problem determination. Advocate pattern going forward.
- Semantic Interoperability: Need a shared understanding of "MRSA" and "Positive for MRSA"
  - Different systems might have different codes in context (Lab Results, Billing, different abbreviations, Identification of an organism vs. a test result)
  - Common terminology helps with a standardized identifier (LOINC)
  - What is "positive"? Value out of reference range? "True/False" flag? Varies by system.
  - Today the logic to handle for the variety of systems is embedded in service logic
  - Public SOA World ➔ Runtime Services / Terminology Server w/ Synonymous terms/codes
  - Semantic Interoperability eased because of common use of "source of truth"

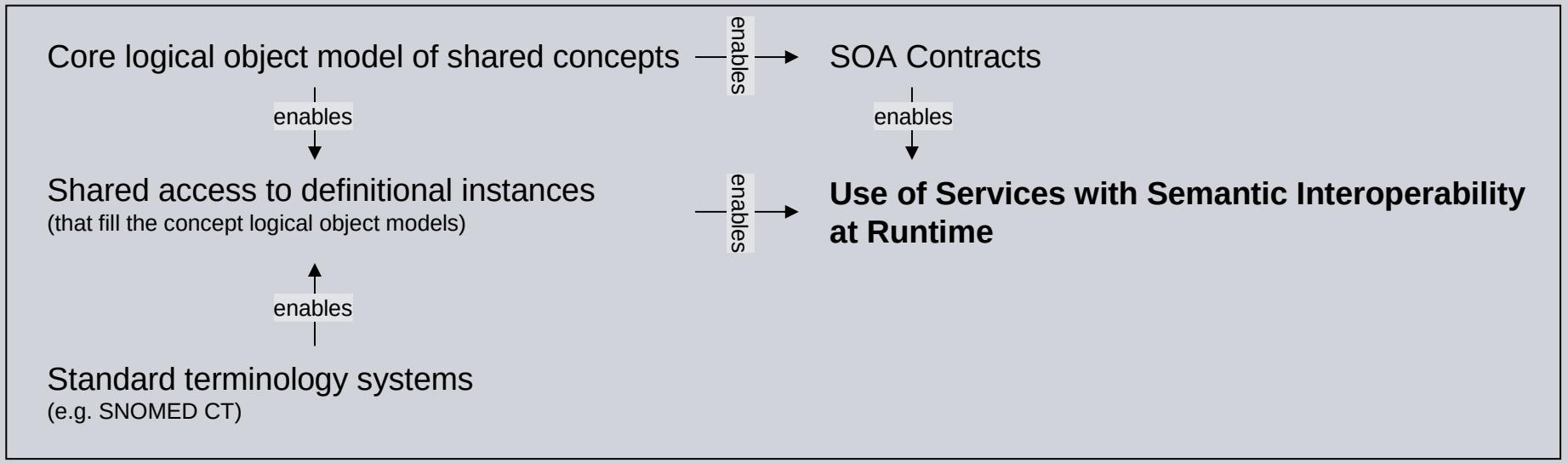
SIEMENS

Service and Event Design Considerations (cont.)

- Location of Interpretation Logic
  - Where to put logic for identifying “Positive MRSA”? In the business service or process post-service-response?
  - Issues include performance, isolation of concerns, consolidation of knowledge
  - Optimizing performance of service call and listener creates a knowledge distribution, and, therefore, increased maintenance effort.
- Monitoring the patient for continued diagnosis of CDIFF, MRSA, VRE has similar design pattern considerations for the Event(s) as for the service
  - Subscribe to all new result events for the patient and process “has MRSA”? Is positive? In logic at the event content return?
  - Or put the MRSA definition (and “is positive?” definition) in the Subscription handler?
- We chose “Get all MRSA results”
  - Chose to put parameters up to disease identification, but short of result value in the service call and event subscription
  - Value (positive) determined in the evaluation logic (in the workflow)

SIEMENS

Semantic Interoperability Enablers



- Enablers are not new, just necessary
- Historical Examples:
  - Healthcare Industry standards: HL7, DICOM (one of the reasons DICOM succeeded in promoting a higher degree of interoperability was its stronger data model as opposed to HL7 2.x)
  - IHE Profiles concentrate not only on the standards employed but the data exchanged to provide for interoperability
  - Other industry standards in banking, insurance, etc.
- Standard Terminology Systems alone provide only a base level of interoperability, do not generally cover necessary operational attributes that must be common for meaningful semantic interoperability

SIEMENS

Right Patients Identified…. Now what?

- The goal is to take the right action, quickly, to achieve patient care benefits:
  - Order the right meds, Move the right patient's to the right place, Alert the right staff alerted for the right reasons,
  - Efficiently for the staff – keeping the signal to noise ratio high so they don't begin to ignore alerts, and
  - Efficiently for the execution and maintenance of the software – keeping concerns isolated, not repeating complex logic in multiple places
- This is the orchestration of macro flows across time, domains, and systems
- Requires "Action" Services
  - Place Alerts: general and by staff (housekeeping, caregivers, by worklist, by beeper, by …)
  - Place Patient on census (by location, by staff)
  - Evaluate / suggest patient transfer
  - (Suggest) Place Order for Medication(s)

“Action” Services – Design Consideration – Routing to Staff

SIEMENS

- Who is the right staff? Depends on Organizational and Encounter information
- Infectious disease logic contains staff to be notified
  - Housekeeping, Patient location ward staff, All care giving staff
  - But must take care not to ‘SPAM’ unnecessary staff
- How much of that info is contained in Encounter? In Organizational structure?
- Where is the logic to decide who to route to?
  - Encapsulate staff member logic within alert service and maintain it there as well as staff/org catalogs?
    - NO! Duplicate, specialized logic, but very course-grained service
  - Infectious disease logic does routing. We chose this option. Base Services we might use include:
    - Get Patient Encounter and Encounter History information service (to pull patient location, medically responsible department, medically responsible staff)
    - Get all patient locations for this encounter service
    - Get all medical staff for this encounter service
    - Get org structure for this Medically Responsible Department
    - Get Staff assigned to these patient locations
    - Get Staff Contact/Alert info (for beepers, etc)
    - Put patient on Patient Location Assignment census with alert

SIEMENS

Design Considerations - isolation of concerns

- Semantics across systems: Manage the “Rights”- Right patient, Right med, Right time, Right dosage, Right route
  - Patient ID, Encounter ID, Medication ID,
  - Same issues as results – public ID schemes, Source of truth for meds
- Isolation of concerns
  - Infectious disease is one part of the patient experience but is not the only thing.
  - Place order, transfer patient need to be considered in context of other things.
  - Suggest place order but have place order be a conscious decision
    - Execute full OE processing (allergy checking, drug-drug interaction checking, cancel other orders checking, etc.)

SIEMENS

5 Years Ago and the Evolution of SOA

- Rich versus Thin
  - Trending towards richer messages - now have multiple consumers over time
  - Reduce callbacks, richer events will do so
- Adoption of technical standards
  - From Proprietary RPC mechanisms to JMS and Web Services
  - Service Metadata Repository (Design-time and Runtime)
  - XSD Schema-driven contracts
- Granularity of events and services
  - Tailored services for rules engine and workflow engine
  - Move toward reusable services - Refactor multiple similar services into single
- Content standards
  - HL7 V2 messages tended to be too large and with a weak model
  - Tailored messages specific to the domain of the service in expected use cases
  - Migration towards information model inspired by HL7 V3 RIM
- Semantic interoperability
  - Central Concept Server with run time services including terminology cross walk services

SIEMENS

In summary

- This is what we learned
  - Principles of SOA have been in place for many years; the technology standards are evolving and solidifying.
  - Content standards and semantic interoperability are still evolving
  - Design tradeoff will continue to affect services definition; there is no one-size-fits-all
  - Healthcare poses significant challenges in domain and semantic handling
  - Operational challenges for use analysis (how many consumers, how consume, etc) pose interesting design challenges
- This is what we are doing or are recommending
  - Public Facing Domain Model – RIM based
  - Centralized Source of Truth with run time services and vocabulary support including cross walk services
  - Granularity of Services and Events at the native business domain's business level (CRUD + business "state" logic (new result, revised result, etc)).
  - Richness of Events at the level that informs the event only (new patient includes patient identity but not full demographics, for example)
  - Richness of Services at two granularities: minimum and full details (get a list of problems is simple list – code, description, update timestamp, get a rich list includes all detail)
  - Consolidation of logic for ease of maintenance even if that might imply larger payloads or less listening efficiency

Questions?

SIEMENS