The case for service oriented architecture in realising trusted, interoperable, pan-European egovernment services.

Stephen McGibbon
Microsoft EMEA
Tel. +445511490070
Email. *stephenm@microsoft.com*

Abstract: This paper seeks to broadly introduce the concept of Service Oriented Architecture and to encourage the debate about i) its utility for realising trusted, interoperable, pan-European egovernment services and ii) whether Europe might not be better to more aggressively pursue work on defining messages and contracts to better achieve interoperability.

1 Introduction

At the EGOV 2004[1] conference two generally accepted barriers to broader deployment of interoperable egovernment solutions were semantic interoperability and trust management. A prevalent view was that to move forward (in either area) required substantial investments most particularly in centralised infrastructure. Within the European context an additional problem was the need to interoperate across member states, each of which may have different systems and policies.
There was some discussion of architectural issues, for example Bellman and Rausch[2] examined the Federal Enterprise Architecture Framework and Figueiredo et al[3] examined the benefits of model driven architecture (MDA), however there was an absence of material on Service Oriented Architecture or service orientation as a concept.

This paper examines how adoption of service oriented concepts, particularly message passing and contract based exchanges, can support the delivery of pan-European interoperable egovernment services. I argue that Service Oriented Architecture provides a possible means of moving forward on both the semantic interoperability and trust management fronts and identify areas where more research could help to achieve this.

This paper seeks only to broadly introduce the concept of Service Oriented Architecture and does not seek to replicate the depth of design advice or definition available readily elsewhere. Instead I seek to encourage the debate about i) its utility for realising trusted, interoperable, pan-European egovernment services and ii) whether Europe might not be better to more aggressively pursue work on defining messages and contracts to better achieve interoperability as current steps to promote

[1] Third International Conference, EGOV 2004 Zaragoza, Spain, August/September 2004
[2] R Traunmüller (Ed.) : EGov 2004, LNCS 3183, pp48-56, 2004.
[3] R Traunmüller (Ed.) : EGov 2004, LNCS 3183, pp260-265, 2004.

interoperability might actually be largely unnecessary with a service oriented approach.

2 Architecture & Service Orientation

2.1 Architecture

The definition of an architecture used in ANSI/IEEE Std 1471-2000is: "the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution."

2.2 Service Orientation

One can think of service orientation as an approach that views systems as being comprised of service providers, and that these service providers expose their capabilities through published interfaces.

1. Loose Coupling

To interact with a service it is generally sufficient to know only the specification of this interface. The details of how the service is actually delivered are thus abstracted from the user who instead relies upon a contract that specifies the service being delivered. This separation between the interface and the implementation is fundamental to the service model.

2.3 The Tenets of Service Orientation

Microsoft[4] has identified four design principles or “tenets” of service orientation:

- Tenet 1: **Boundaries Are Explicit**. Services interact through explicit message passing behind the boundaries. We make no assumptions on the space behind the service boundaries. Crossing service boundaries can be costly (for example, you may need to span geography, trust boundaries, or execution environments). We explicitly opt in to service invocation, by formally passing defined messages between services. The explicit boundaries allow us to formally express implementation independent interaction — we can be agnostic to platform, middleware or coding language choices used to implement other services.
- Tenet 2: **Services Are Autonomous**. Services behave reasonably as independent entities. We make no assumptions on the space between the service boundaries. There is no presiding authority in a service-oriented environment. Services are independently deployed, versioned and managed. The topology in which a service executes can and will evolve. The service should expect that peer services can and will fail and that it can and will received malformed or malicious messages. Services should be built not to fail, using (for example) redundancy and failover techniques.
- Tenet 3: **Services Share Schema and Contract, Not Class**. Services interact solely on their expression of structures using schema and behaviours using contract. The service’s contract describes the structure of messages and ordering constraints over messages. The formality of the expression allows machine verification of incoming messages, which allows us to protect the

[4] Specifically, Don Box of the Microsoft Indigo team

service's integrity. Contracts and schema must remain stable over time, so building them flexibly (for example, through use of xsd:any in schema) is important.

- Tenet 4: **Service Compatibility Is Based on Policy**. Both service providers and service consumers will have policies — operational requirements — for interactions across boundaries. A simple example of provider side policy is that a service may require that the invoker have a valid account with the service provider. From the consumer side, an organization may require that service invocations across the Internet be encrypted, so it will only use services that offer the capability of bidirectional security-enhanced data exchanges. Services express their capabilities and requirements in terms of a machine-readable policy expression. Policy assertions are identified by a stable, globally unique name. Individual policy assertions are opaque to the system at large; services must simply be able to satisfy each other's policy requirements.

Communications at the boundary of your service should support the above tenets. These tenets require a formal expression of schema, contract and policy between participating services in the service-oriented environment. While ad hoc mechanisms can be developed for expressing the boundaries of services, such mechanisms are limited to the scope of influence of the inventor. For example, if you develop a system that expresses schema and contract in a manner which is recognizable only in your department, you will have prevented your service from ever being usable outside of your department. Millard[5] found "a clear and strong link between reorganizing government back-offices and the electronic public services experienced by users". The strategies he identifies are all consistent with the adoption of a Service Oriented Architecture, and indeed would be made substantially easier to achieve.

To fully realize the benefits of service orientation, the expression of the service boundary must be as broadly adopted and interoperable as possible. The industry has clearly chosen WS-* protocols propagated in SOAP messages as the interoperability standard for services. Service orientation practically applied requires Web services, SOAP messages and use of the WS-* protocols.

3 Service Oriented Architecture

Service oriented architecture maps the capabilities and interfaces of services so that they can be orchestrated into processes. Such a process will itself be a service and will therefore expose its own capabilities through a published interface thus providing a level of abstraction over its constituent sub-services.

3.1 Composability, Orchestration & Abstraction

Humans rely on abstraction to reduce complexity. Existing governmental and business processes all abstract services from the various departments and orchestrate them together to produce meaningful work – one department will not worry about how another organizes their internal operations, instead relying on definitions of input, output and service level agreement. Service oriented architecture parallels this approach and is therefore particularly suited to the development and delivery of pan-European interoperable egovernment services.

[5] R Traunmüller (Ed.) : EGov 2004, LNCS 3183, pp363-370, 2004.

Service oriented architecture provides an important means of simplifying complex processes through service abstractions in distributed systems architecture. Services are often said to be composable, i.e. they can be aggregated to produce more complex services. The composition may use workflow techniques through a process called orchestration, or may be “hand-stitched”. Verginadis et al[6] have documented the advantages of using recurring workflow blocks in modelling egovernment service provision.

Choreography can guarantee interoperability.

Services are sometimes thought of as having “fractal dimension” as a result of this property that composed services are themselves abstracted to a new service. The important point is that the level of complexity being managed appears the same despite the scale, (in egovernment terms we might think of local, departmental, ministry, central government, EU for example). I would suggest that the problems identified by Leenes & Svensson[7] result when this is not the case.

3.2 Extensibility

In just the same way that a single implementation of a service is hidden from the user as described above, so too may multiple implementations - the service can be versioned to support revisions in process, such as when tax rules change with a budget, or to support different rules for pension contributions for nationals, resident workers etc.

A service may, and typically will, support multiple message versions and Messages themselves remain compatible. The service may even be totally refactored, whereby its internal operation is redesigned or rewritten but the interfaces are preserved.

3.3 Contracts

A contract is a specification of the way the consumer and provider of a service will interact. It specifies the format of the request to, and response from, the service. A service contract may have a set of prerequisites that typically incorporate satisfactory authentication and authorisation. The contract may also specify quality of service (QoS) levels or SLAs (Service Level Agreements) such as the amount of time it takes to execute a service method, guarantee of service etc.
Contracts can be thought of as defining the allowed sequence and format of messages exchanged amongst services.

3.4 Messages

A service is a program that communicates by exchanging messages. Put another way, a service is a unit of application logic whose interface is defined purely by the messages it will accept and send.

Message passing as a means of loosely coupling systems to promote interoperability is not a new concept. Many will be familiar with an organisation called SWIFT[8], however not everyone is aware of the volume of SWIFT traffic – trusted,

[6] R Traunmüller (Ed.) : EGov 2004, LNCS 3183, pp483-488, 2004.
[7] R Traunmüller (Ed.) : EGov 2004, LNCS 3183, pp496-502, 2004.
[8] SWIFT is the financial industry-owned co-operative supplying secure, standardised messaging services and interface software to 7,650 financial institutions in over 200 countries. SWIFT's worldwide community includes banks, broker/dealers and investment managers, as well as their market infrastructures in payments, securities, treasury and trade.

interoperable transactions that cross both organisational and national boundaries. In 2004 some 1,514,420,079[9] messages were passed across SWIFT's FIN[10] network in Europe alone:-

- SWIFT defines the format and semantics of the messages
- SWIFT operates a PKI and certifies members
- SWIFT does not mandate which core business systems a member should use, or how a bank should organise its internal processes and operations.
- SWIFT does require members to operate a common process for handling SWIFT messages though, and provides the training, accreditation and support to achieve this.

This example of successful delivery of loosely coupled interoperable pan-European banking transactions based on the transfer of defined messages can and should inform the current development in Europe around interoperable e-Government services.

In the emerging service oriented architecture model composed of web services specifications such as XML, XSD, WSDL, UDDI, WS-* (WS-Security and WS-Policy) are used to deliver :-

- A standard syntax in which information from all systems could be unambiguously expressed
- Standard semantic models so that organizations could express their business practices in a consistent language
- Standard protocols so that information could be passed across boundaries between operating environments and between organizations
- A standard means for binding behaviour to business documents

4 Trust

Microsoft regards Trustworthy Computing as more than simply security. In order for egovernment services to be regarded as trustworthy they will also have to be consistent, reliable, transparent, and respect the citizen's privacy.

Whilst this poses challenges for individual services, there seems to be little prospect of achieving consistency across egovernment systems delivering services throughout Europe unless something broadly similar to the service oriented concepts discussed are adopted.

Regarding both authentication and authorisation as common aspects of a pan European eGovernment service oriented architecture, and adopting emerging aspect oriented techniques offers, in the authors opinion, sufficient grounds to expect that progress will be made here too. As discussed earlier, such common aspects would then be bound in the contract specification as pre-requisites for the orchestration to proceed.

Existing investments in PKI would allow messages between administrations to be trusted much as SWIFT does already. This would then allow new and novel approaches to the enrolment of citizens and subsequent authorisation of egovernment transactions.

[9] http://www.swift.com/index.cfm?item_id=43626

[10] FIN services are SWIFT's core store-and-forward messaging services. FIN enables over 7,500 financial institutions in more than 200 countries to exchange financial data securely, cost effectively and reliably. In 2000, FIN carried over 1.2 billion messages. Peak days are currently running above 9 million messages. (Source : http://www.swift.com/index.cfm?item_id=3184)

Common standards will be required for auditing, records management and disposition, and change management. Again the author sees hope in the success of SWIFT – if the EU sought to fulfil some of the functions for egovernment services that SWIFT fulfils for interbank services then perhaps progress here can also be swift.

5 Impact on design and delivery of egovernment services in Europe

It is possible to envisage a pan-European trusted interoperable egovernment service infrastructure that delivers interconnected loosely coupled, coarse-grained services through use of such a Service-Oriented Architecture approach.

To move towards this it is necessary, in the author's view, for a more service oriented view of egovernment services to be adopted than is currently the case.

Nadhan identified[11] (and discusses solutions to) the following eight areas that typically present challenges when deploying SOA:-

1. Service identification. What is a service? What is the business functionality to be provided by a given service? What is the optimal granularity of the service?
2. Service location. Where should a service be located within the enterprise?
3. Service domain definition. How should services be grouped together into logical domains?
4. Service packaging. How is existing functionality within legacy mainframe systems to be re-engineered or wrapped into reusable services?
5. Service orchestration. How are composite services to be orchestrated?
6. Service routing. How are requests from service consumers to be routed to the appropriate service and/or service domain?
7. Service governance. How will the enterprise exercise governance processes to administer and maintain services?
8. Service messaging standards adoption. How will the enterprise adopt a given standard consistently?

Progressing in these areas will bring the goal of interoperable pan-European egovernment services considerably closer to realization.

6 Conclusion

This paper has proposed that a service oriented approach would make the realisation of pan-European, interoperable and trusted e-government services substantially easier. A parallel was drawn with the SWIFT network in the world of banking, and a set of areas for future work was identified.

An egovernment service would then be generally implemented as a discoverable software entity that interacts with applications and other services through a loosely coupled (often asynchronous), message-based communication model.

I opened this paper by stating that I hoped to encourage the debate about whether current steps to promote interoperability might actually be largely unnecessary with a service oriented approach, and whether Europe might not benefit from a more aggressive pursuit of the work needed to define messages and contracts to better achieve interoperability in egovernment services. I hope that the arguments laid out encourage such a debate and welcome comments.

[11] Easwaran G. Nadhan, Service-Oriented Architecture: Implementation Challenges, http://msdn.microsoft.com/architecture/soa/default.aspx?pull=/library/en-us/dnmaj/html/aj2soaimpc.asp