Towards a Design Approach
for an Effective System
Evolution of a Large Electronic Archive Information System

Quyen L. Nguyen
NARA ERA
qnguyen@nara.gov
Agenda

- Background
- Problem Statement
  - System Evolution
  - Evolving the System
- Service Decomposition
- Service Composition
- Conclusion
Agenda

- Background
- Problem Statement
  - System Evolution
  - Evolving the System
- Service Decomposition
- Service Composition
- Conclusion
Electronic Records Archiving Challenges

- **Variety**
  - Records come from various application domains: health care, education, space exploration, census statistics, etc.
  - Different types of records: MS Office, database files, geospatial images, multimedia objects

- **Obsolescence**
  - To-be ingested records come from obsolete or old versions of applications and computing platforms

- **Volume**
  - Expected large amount of records: terabyte, petabyte, or exabyte
  - Volume growth is of exponential rate, in terms of total size and number of digital objects
ERA System Requirements
Electronic Records Archive

- **Extensibility**: record types, data types, and services could be added without extensive redesign.
- **Availability**: key functions must be available more than 99%; no single point of failure.
- **Security**: protection of system and its assets.
- **Evolvability**: new technologies could be inserted using standards APIs and interfaces.
- **User Friendly**: browser interface, intuitive, 508 compliance.
- **Scalability**: adapt to record volume and user community growth.
OAIS Functional Model
Open Archive Information System
Agenda

- Background
- Problem Statement
  - System Evolution
  - Evolving the System
- Service Decomposition
- Service Composition
- Conclusion
Local Evolution

- **Domain**
  - Higher scalability and throughput to accommodate the growth of digital record volume and user community.
  - Changing business rules
  - Archival science and concept of “essential characteristics” of digital record are still evolving

- **Experience**
  - Experienced users will demand enhanced search and discovery capabilities
  - Web 2.0: forum, tagging, interest groups, etc.
  - Soon, there will be a demand to access digital records not only on workstations but other devices such as mobile phones, or Wi-Fi enabled handheld devices.

- **Process**
  - New techniques and methodologies must be inserted to the system.
Evolution Relativity

- Upper timeline shows evolution of the system itself.
- Lower timeline shows evolution of external systems that created to-be archived data.
- Note the lags between the two timelines (several years).
- Challenge for WAIS: evolving itself to use current technologies of time $T_a$ in order to provide long-term access to data born out of technologies at $T_c$ time.
Data-Centric View of Evolution

- Ingested data
  - No longer active records
  - May need transformation before storing
- Stored data
  - Provide access to data using current technology
  - Data transformation and migration for preservation purposes
- Business data
  - Created by WAIS itself to govern the operations inside the system
Problem and Related Work

- What approach to take to face the challenges of evolution in terms of:
  - Maintaining the System
  - Evolving the System

- Christian Saul and Fanny Klett had an SOA-based framework to model a digital preservation system.

- Street and Gomaa provided guidance to maximize reusability of SOA systems via architectural constructs and patterns, service interface, service logic, service messaging, service discovery, and service differentiation.

- Papazoglou suggested leaning towards coarse-grained and high-level services so that they are more useful to business applications.
Agenda

- Background
- Problem Statement
  - System Evolution
  - Evolving the System
- Service Decomposition
- Service Composition
- Conclusion
SOA Approach

- **External Systems**
  - Adapters (protocol binding and data transformation)

- **Ingest**
  - SIP
  - Transfer Processing
  - Ingest Processing

- **Access**
  - Query Federation
  - Orders
  - Result Sets

- **ESB**
  - Commands
  - Queries/Responses

- **Preservation**

- **Data Management**

- **AIP**
  - AIP Routing

**Key Components**

- **PRODUCER**
- **CONSUMER**
Service Decomposition Scheme (SDS)

- Start with the set of high-level services $S = \{\text{Ingest, Storage, Access, Data Management, Preservation Planning, Administration}\}$.

Call $\text{decompose}(S)$.

$\text{decompose}(X)$:
For each service $s$ in $X$, do:
  - If $s=<<\text{tool>>}$, stop.
  - Else
    - Decompose $s$ into a set of components $R$.
    - Call $\text{decompose}(R)$.
  Endif.
Endfor.
Stopping criterion is <<tool>>: corresponds to the lowest level of the service in the decomposition design process

What constitutes the stereotype <<tool>>?

- A service is considered to be a “tool” if there is an implementation provided by a Commercial-off-the-shelf Software (COTS) or Free Open Source Software (FOSS)
  - In digital archiving and preservation, such software packages exist

For file format identification:

- DROID (Digital Record Object Identification) from UK National Archives
- JHOVE (JSTOR/Harvard Object Validation Environment)
- APT (Archival Processing Tool) from Georgia Tech.

For format transformation:

- Xena Digital Preservation Software from Australia National Archives
- Oracle Outside In
Example

\textit{decompose}(Ingest):
Ingest is not a \textless \textless tool\textgreater \textgreater .
Decompose Ingest into a set of components \( R = \{ \text{FormatIdentify, ExtractMetadata, BuildAIP, ValidateIngestData} \} \).
Call \textit{decompose} (FormatIdentify).
FormatIdentify is not a \textless \textless tool\textgreater \textgreater .
Decompose FormatIdentify into a set of components \( R = \{ \text{DROID, JHOVE, APT} \} \).
DROID is a \textless \textless tool\textgreater \textgreater , stop.
JHOVE is a \textless \textless tool\textgreater \textgreater , stop.
APT is a \textless \textless tool\textgreater \textgreater , stop.
Call \textit{decompose} (ExtractMetadata)….
Agenda

- Background
- Problem Statement
  - System Evolution
  - Evolving the System
- Service Decomposition
- Service Composition
- Conclusion
Service Composition

- Provide service interfaces to the applications
- Facilitate evolvability within a composition
- Use of BPEL
- Ability to modify logic in the BPEL script to
  - Swap in and out components as they evolve
  - Add new <<tool>> components to the process
Maintaining WAIS
Web-based Archive Information System

- Maintaining the system is tantamount to
  - Monitor evolution of existing <<tool>> components
  - Monitor evolution of current applications and their associated <<tool>> that are utilized during the archiving and preservation process

- What about <<tool>> that do not exist yet?
  - The service decomposition helps in defining interfaces for new <<tool>> components
  - These <<tool>> components will be researched and developed by the community
Ingest Service Composition Example
File Format Identification

1. Identify(digitalObject)
FormatIdentify

2. indentify(digitalObject)
DROID

3. validate(digitalObject)
JHOVE
Preservation Service Composition Example
Preservation Manager

1. preserve(preservationPlan, AIP)

2a. [strategy=transform]
   <<service>>
   Transformation

2b. [strategy=UVC]
   <<service>>
   UVCSpecBuilder

2c. [strategy=MV]
   <<service>>
   MultiValenceExtractor

3a. [transform=pdf]
   <<tool>>
   PDFConverter

3b. [transform=xml]
   <<tool>>
   XMLConverter
Access Service Composition Example

Search Manager

1. search(criteria)

<<service>>
SearchManager

2a. search(criteria)

<<service>>
ContentSearch

3a. [type=text]
<<tool>>
TextualSearch

3b. [type=multimedia]
<<tool>>
MultimediaSearch

2b. search(criteria)

<<service>>
MetadataSearch
Current Issues

- Success depends on interests, research and development of <<tool>>
- Requires collaboration between digital libraries, national archives, and research institutions
- Could this concept of <<tool>> be applied to other SOA systems?
  - In some sense, a <<tool>> can be considered as a standardized part of a large machine that you can easily get from any “manufacturer”
  - Is this going back to the dream of software engineering to have ready-made components?
Summary

- Presented the evolution characteristics of a WAIS system
- Proposed an SOA-based approach for Evolvability:
  - Service Decomposition Scheme with <<tool>> concept
  - Service Composition using BPEL with application to WAIS
- Showed how the <<tool>> concept can help in the maintenance of this SOA system
- Presented some issues with the current solution
Thank You
References


