



## Measuring Reliability of Applications Composed of Web Services

March 22, 2007

**Hangjung Zo**

Information and Communications University

### OUTLINE

## Outline

- > Introduction
- > Web Services Reliability
- > Solution Methods
- > Application
- > Results
- > Implications and Limitations
- > Q & A

## Motivation

- > Limitations of traditional development approaches
- > Need for more agile approaches
- > **Web Services** as an alternative basis for the rapid development
- > Concerns about performance of applications developed using Web services

## Application Composition using Web Services

- > Selection
  - Selection criteria
- > Assembly
  - Orchestration
  - Choreography
- > Evaluation

## Web Services Selection Criteria

- > Functionality
- > Performance
- > Non-functional criteria
  - Security
  - Reliability
  - Cost
  - Vendor reputation

## Research Objectives and Method

- > Devise an approach to measure reliability of applications composed of web services
- > Develop a method for selecting appropriate Web Services using reliability metrics
- > Use of the **Design Science Methodology** to provide a robust solution to the problem

## Reliability

- > Probability that the software will be functioning without **failure** under a given environmental condition during a specified period of time (Shooman 1984, Xie 1991)
  
- > Measuring Reliability (Musa 1999)
  - Time-based metrics
  - Failure-based metrics
  
- > Web Services: **RPC** method over the Internet
  - Reliability measured as a percentage of failed invocation instances

## Computing Process Reliability

- > Reliability for **business process** based upon
  - Reliability of web services for individual tasks
    - Web service reliability
    - Network reliability
  - Structure of business process
    - Sequence
    - Parallel (AND split – AND join)
    - Conditional (XOR split – XOR join)
    - Simple loop
    - Dual loop
  - Redundancy of support for individual tasks

## Combining Reliability Metrics

- > Reliability for **business process** based upon
  - Reliability of web services for individual tasks
  - Structure of business process
  - Sequential System
  - Parallel System (AND split – AND join)
  - Conditional System (XOR split – XOR join)
  - Loop system (Simple and Dual loop)
  
- > Consider Reliability at **the task level** and **redundancy support** for a task should be accounted for

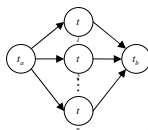
## Combining Reliability Metrics (Cardoso et. al.)

**Sequence**



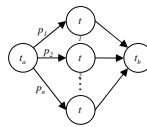
$$R = R(t_i)R(t_j)$$

**Parallel**  
(AND split – AND join)



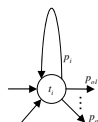
$$R = R(t_a)R(t_b)\prod_i R(t_i)$$

**Conditional**  
(XOR split – XOR join)



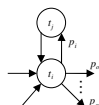
$$R = R(t_a)R(t_b)\sum_i p_i R(t_i)$$

**Simple loop**



$$R = \frac{(1 - p_i)R(t_i)}{1 - p_i R(t_i)}$$

**Dual loop**



$$R = \frac{(1 - p_i)R(t_i)}{1 - p_i R(t_i)R(t_j)}$$

## Redundancy-based Reliability

- > Reliability at the task level affected by redundant support, if available

$$R_k = 1 - \prod_{j \in S} (1 - R_j)$$

## Problem Characteristics

- > Single Objective: Reliability
- > Combinatorially Explosive Search: Large Solution Space
  - 10 tasks, 50 web services  $\sim 10^{17}$  possible solutions
- > Non-deterministic objective functions
- > Optimization techniques not feasible
- > Exhaustive search impractical for large problems
- > Heuristic Approaches
  - Genetic Algorithms
  - Simulated Annealing/Tabu Search

## Genetic Algorithms (GA)

- > An adaptive search technique based on principles of natural evolution and heredity (**natural selection** and reproduction)
- > Appropriate for solving a problem in a space that is large to be searched (**not smooth, unimodal**, or well-understood)
- > Evaluation (fitness) function is **noisy** and does **not require a global optimum** to be found
- > NSGA (Nondominated Sorting Genetic Algorithm)
  - Srinivas and Deb (1994)

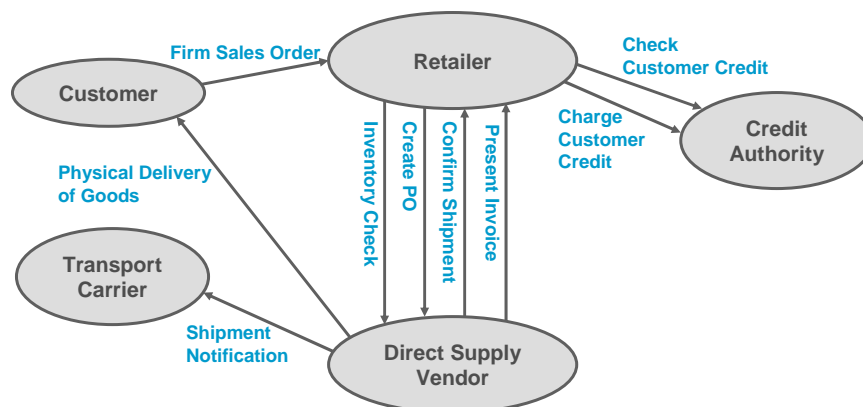
## Implementing GA

- > Representation
  - **Encoding potential solutions** to the problem
  - Binary Encoding
- > Initialization
  - **Generating an initial population**
  - Randomly Generated
- > Evaluation and selection
  - **Rating solutions in terms of their fitness** and selecting the population for the next generation
- > Genetic Operators
  - **Crossover**
  - **Mutation**
- > Parameters
  - Population size, Number of Elitists, etc.

## Direct-to-Customer Drop Ship Retail Model

- > A retail business model that delivers products directly to a customer from a vendor (UN/CEFACT and OASIS 2001)
- > Typical business model in the online retailing area
- > Includes Order Management, Order Fulfillment, and Payment

## Direct-to-Customer Drop Ship Retail Model



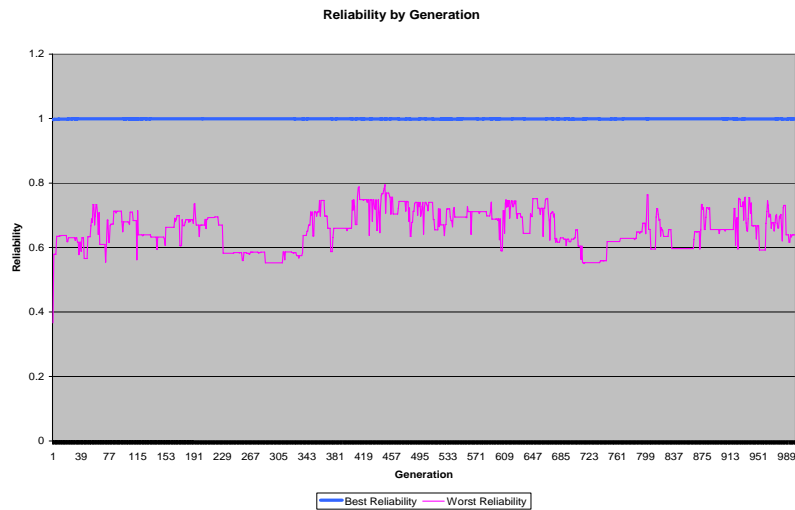
## Drop-Ship Example

Sub-Process	1	2	3	4
No of Tasks	4	14	6	10
Web Services	24	75	33	57
Total Solutions	$3.32 \times 10^5$	$1.78 \times 10^{26}$	$1.29 \times 10^9$	$3.62 \times 10^{17}$
Feasible Solutions	$1.05 \times 10^5$	$4.50 \times 10^{17}$	$8.20 \times 10^6$	$1.56 \times 10^{12}$

## GA Implementation Parameters

- > Chromosome Size: 99 to 999 bits
- > Population Size: 1000
- > Evaluation Function: Symbolic expression of aggregate reliability
- > Crossover rate: 0.8
- > Mutation rate: 0.8
- > Number of generations: 300, 1000
- > Java implementation

## GA Results



ICU

## Implications for System Designers

- > For the selected example, good solutions can be obtained with relatively **little search effort**
- > Wide range in quality of solutions observed
- > Provides a basis for measuring reliability of applications composed of Web Services
- > Provides **an opportunity to analyze an extremely large search space** for good solutions quickly

ICU

## Limitations and Future Directions

- > Focuses merely on [reliability](#) to select vendors and Web services
- > can be extended to [Multi-Objective selection problem](#)
- > [Further validation of the approach is necessary](#) with other business process
- > Employs [a conceptual and abstract business model](#) (direct to customer drop ship retail model) as an application of the model
- > [Real-world case](#) would strengthen its credibility

## Questions and Comments?

