A Distributed Replication Strategy Evaluation and Selection Framework for Fault Tolerant Web Services

Zibin Zheng and Michael R. Lyu

Department of Computer Science & Engineering
The Chinese University of Hong Kong
Hong Kong, China

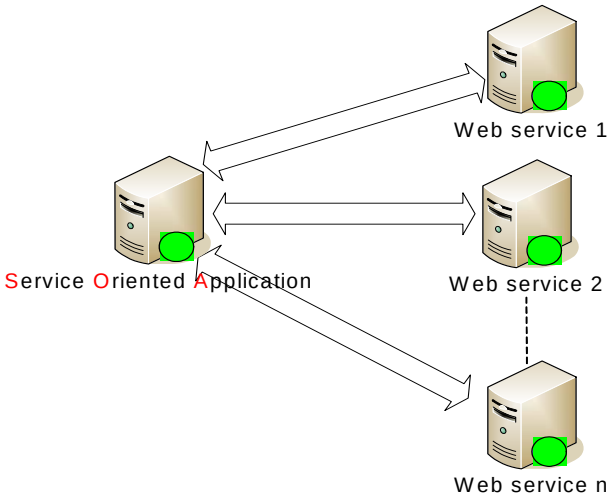
ICWS 2008, Beijing, China, 24 September, 2008

Outlines

1. Introduction
2. Distributed Evaluation Framework
3. Fault Tolerance Replication Strategies
4. An Optimal Strategy Selection Algorithm
5. Experiments
6. Conclusion

1. Introduction

- Web services are becoming popular.
- Reliability of the service-oriented applications becomes difficult to be guaranteed.
  - Remote Web services may contain faults.
  - Remote Web services may become unavailable.
  - The Internet environment is unpredictable.



1. Introduction

- Traditional software reliability engineering
  - Fault Tolerance is a major approach for building highly reliable system.
  - Expensive.
- Service reliability engineering
  - Web service with identical/similar interface
  - Less expensive & less time-consuming
- A lot of fault tolerance replication strategies
  - Time redundancy
  - Space redundancy

Which fault tolerance replication strategy is optimal?

1. Introduction

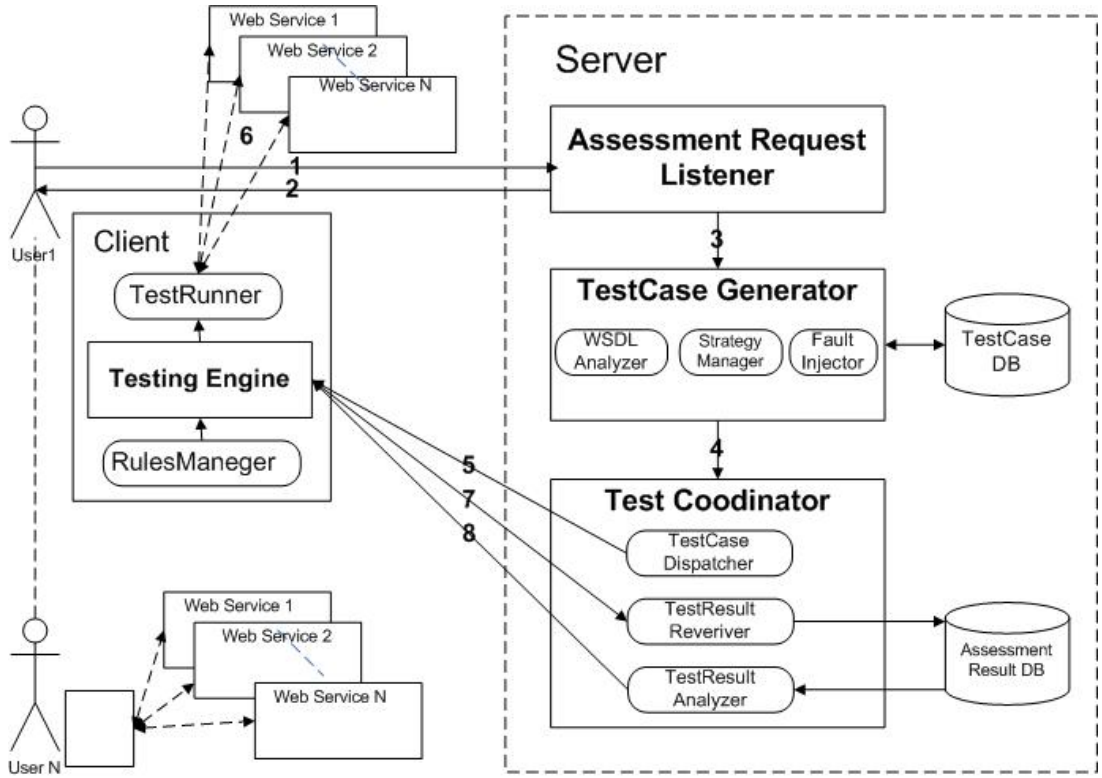
- A Distributed Replication Strategy Evaluation and Selection Framework for Fault Tolerant Web Services.
  - User-collaborated evaluation
    - *YouTube*: sharing videos.
    - *Wikipedia*: sharing knowledge.
    - Sharing evaluation results of target Web services.
  - Evaluation of individual Web service
  - Evaluation of fault tolerance strategies
  - Optimal fault tolerance strategy selection.

2. Distributed Evaluation Framework

- Service users
  - Web service selection
  - Overall performance of target Web services
    - Different locations
    - Long time duration
- Service providers
  - Providing better services
  - Overall performance of their own Web services

- Overall performance of Web services is not easy to be obtained
  - Time-consuming
  - Expensive

2. Distributed Evaluation Framework



1. Evaluation request
2. Load Applet
3. Create test cases
4. Schedule test tasks
5. Assign test cases
6. Client run test cases
7. Send back results
8. Analyze and return final results to client.

- Evaluation results from different locations
- Don’t need good knowledge on FT strategies, test case generation, and so on.
- Don’t need to implement evaluation mechanism.

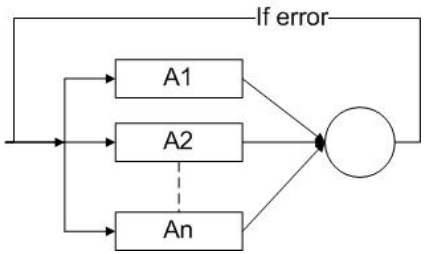
3. Replication Strategies

- Basic replication strategies.
  1. **Active.** The application sends requests to different replicas at the same time and uses the first properly returned response as final result.
  2. **Time.** The same Web Service will be tried one more time if it fails at first.
  3. **Passive.** Another standby Web Service will be tried in sequence if the primary Web Service fails.

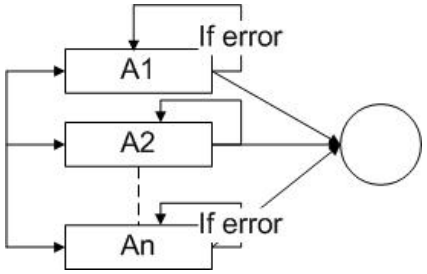
	Active	**Time**	**Passive**
Active	1.Active	4. Active+Time	6. Active+Passive
Time	5. Time+Active	2. Time	8. Time+Passive
Passive	7. Passive+Active	9. Passive+Time	3. Passive

3. Replication Strategies

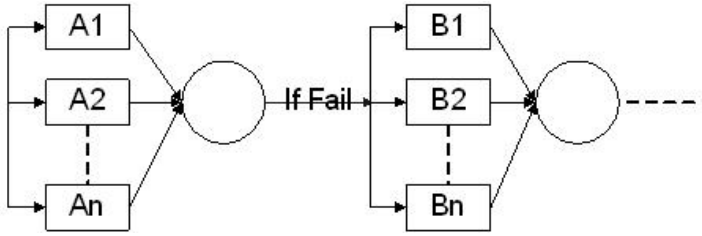
4. Active+Time



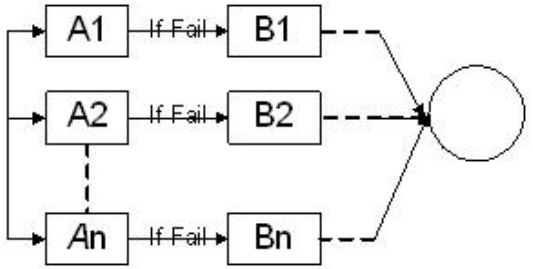
5. Time+Active



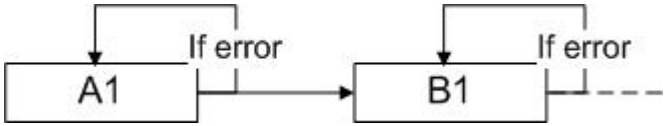
6. Active+Passive.



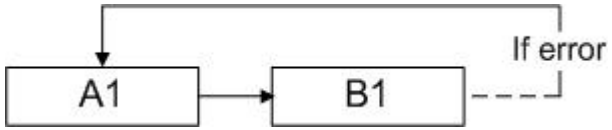
7. Passive+Active



8. **Time+Passive**



9. Passive+Time



3. Replication Strategies

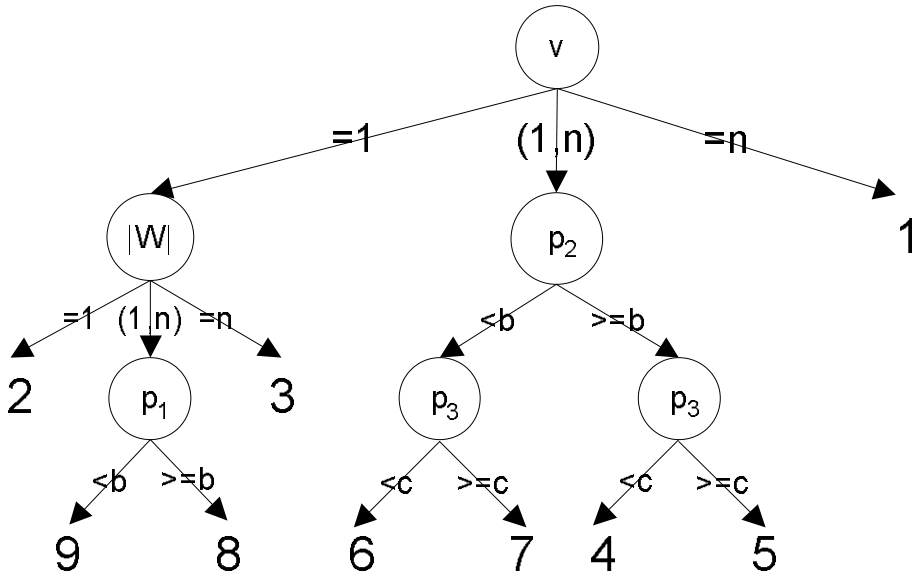
	Formula
$\overset{p}{1}$	$r = 1 - \prod_{i=1}^{n}(1 - r_i);$ $t = \begin{cases} \min\{T_c\} : \lvert T_c\rvert > 0 \\ \max\{T_f\} : \lvert T_c\rvert = 0 \end{cases} ; T = \{t_1, ..., t_n\} = T_c \cup T_f$
$\overset{s}{2}$	$r = 1 - (1 - r_1)^m; t = \sum_{i=1}^{m} t_i (1 - r_1)^{i-1};$
$\overset{s}{3}$	$r = 1 - \prod_{i=1}^{m}(1 - r_i); t = \sum_{i=1}^{m} t_i \prod_{k=1}^{i-1}(1 - r_k)$
$\overset{h}{4}$	$r = 1 - (\prod_{i=1}^{v}(1 - r_i))^m;$ $t = \sum_{i=1}^{m} t_i (\prod_{j=1}^{v}(1 - r_j))^{i-1}; t_i = \begin{cases} \min\{T_c^i\} : \lvert T_c^i\rvert > 0 \\ \max\{T_f^i\} : \lvert T_c^i\rvert = 0 \end{cases}$
$\overset{h}{5}$	$r = 1 - \prod_{i=1}^{v}(1 - r_i)^m;$ $t = \begin{cases} \min\{T_c\} : \lvert T_c\rvert > 0 \\ \max\{T_f\} : \lvert T_c\rvert = 0 \end{cases} ; t_i \in T = \sum_{j=1}^{m} t_{ij}(1 - r_i)^{j-1}$
$\overset{h}{6}$	$r = 1 - \prod_{i=1}^{m}\prod_{j=1}^{v}(1 - r_{ij});$ $t = \sum_{i=1}^{m} t_i \prod_{k=1}^{i-1}\prod_{j=1}^{v}(1 - r_{kj}); t_i = \begin{cases} \min\{T_c^i\} : \lvert T_c^i\rvert > 0 \\ \max\{T_f^i\} : \lvert T_c^i\rvert = 0 \end{cases}$
$\overset{h}{7}$	$r = 1 - \prod_{j=1}^{v}\prod_{i=1}^{m}(1 - r_{ij});$ $t = \begin{cases} \min\{T_c\} : \lvert T_c\rvert > 0 \\ \max\{T_f\} : \lvert T_c\rvert = 0 \end{cases} ; t_i = \sum_{j=1}^{m} t_{ij} \prod_{k=1}^{j-1}((1 - r_{ik})$
$\overset{s}{8}$	$r = 1 - \prod_{i=1}^{u}(1 - r_i)^m;$ $t = \sum_{i=1}^{u}((\sum_{j=1}^{m} t_i (1 - r_i)^{j-1}) \prod_{k=1}^{i-1}(1 - r_k)^m);$
$\overset{s}{9}$	$r = 1 - (\prod_{i=1}^{u}(1 - r_i))^m;$ $t = \sum_{i=1}^{m}((\sum_{j=1}^{u} t_j \prod_{k=1}^{j-1}(1 - r_k)(\prod_{j=1}^{u}(1 - r_j))^{i-1});$

4. Selection Algorithm

- Objective evaluation results of Web services.
- Subjective requirement of service users
  - *t-user*:
    - represents the user requirement on response time improvement of increasing one parallel replica.
    - designed to facilitate the user to make a tradeoff between the response time performance and resource consuming.
  - *f-user*:
    - the failure-rate requirement provided by users.

4. Selection Algorithm

- Determining parallel replica number: *v*.
- Excluding bad performance replicas: *|W|.*
- Determining detailed optimal strategy based on: *p1,p2,p3*.



$$s_i = \frac{t_i}{t_{user}} + \frac{f_i}{f_{user}}.$$

$$W = \{ws_i | s_i \leq a \&\& 1 \leq i \leq n\}$$

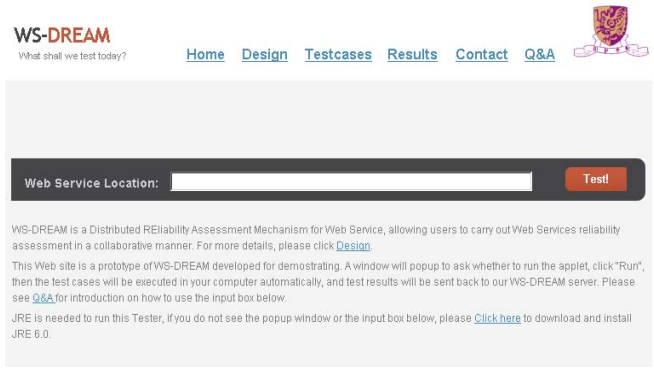
$$p_1 = s_2 - s_1$$

$$p_2 = \frac{1}{v}\sum_{i=1}^{v}(s_{i+v} - s_i)$$

$$p_3 = \frac{1}{v}\sum_{i=1}^{v} f_i$$

5. Experiments

- JDK + Eclipse
- Client-side:
  - Java Applet
- Server-side:
  - an HTTP Web site (Apache HTTP Server)
  - a TestCaseGenerator (JDK6.0 + Axis library)
  - a TestCoodinator (Java Servlet + Tomcat 6.0)
  - a MySQL database (Record testing results)



5. Experiments

1. Evaluating the performance of individual Web Services.
2. Evaluating the performance of different fault tolerance strategies employing the six identical Web services provided by Amazon.
3. Determining the optimal fault tolerance strategy.

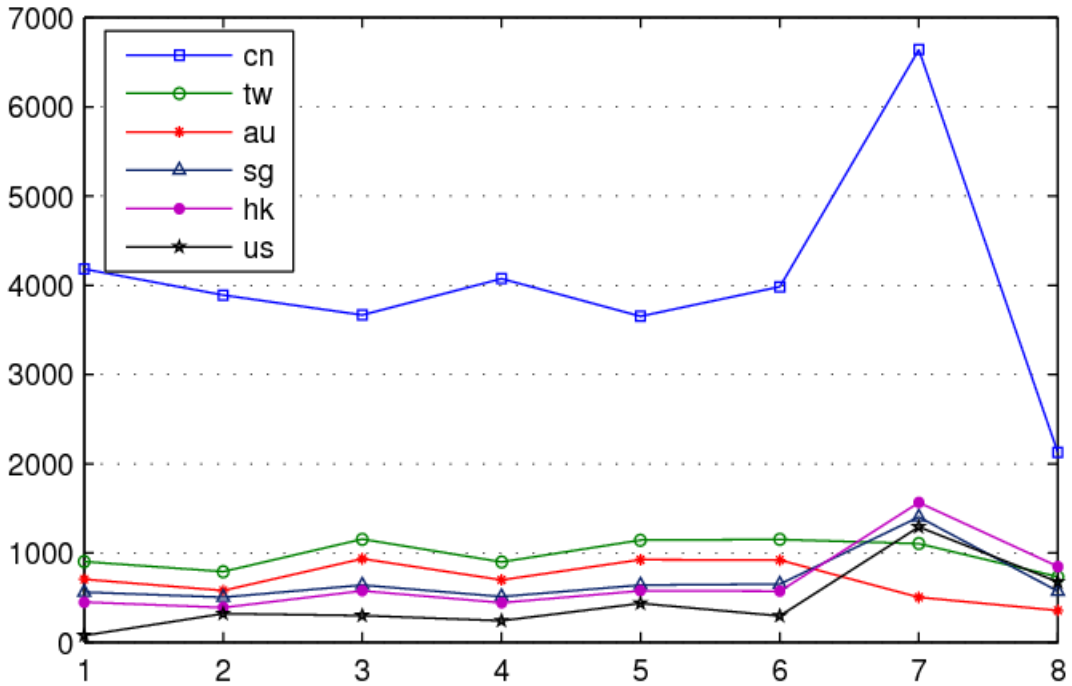
5.1 Results-individual WS

Table 3. Evaluation Results of the Eight Target Web Services

Location		Cases			RTT (ms)				Location		Cases			RTT (ms)			
L	WS	All	Fail	R%	Avg	Std	Min	Max	L	WS	All	Fail	R%	Avg	Std	Min	Max
cn	a-us	484	109	22.52	4184	2348	562	9906	tw	a-us	2470	0	0	902	294	578	4609
	a-jp	482	128	26.55	3892	2515	547	9937		a-jp	2877	1	0.03	791	315	407	5016
	a-de	487	114	23.40	3666	2604	687	9844		a-de	2218	0	0	1155	355	765	4547
	a-ca	458	111	24.23	4074	2539	610	9953		a-ca	2612	5	0.19	899	300	562	4032
	a-fr	498	96	19.27	3654	2514	687	9999		a-fr	2339	0	0	1144	370	734	4813
	a-uk	493	100	20.28	3985	2586	719	9875		a-uk	2647	1	0.03	1150	363	750	5093
	GW	409	337	82.39	6643	2003	2094	9969		GW	1981	35	1.76	1105	1401	343	9844
	GIP	540	32	5.92	2125	1927	531	9781		GIP	2822	60	2.12	732	1270	265	9875
au	a-us	1140	0	0	705	210	500	3782	sg	a-us	1895	0	0	561	353	297	4406
	a-jp	1143	0	0	577	161	406	2594		a-jp	1120	0	0	503	322	250	3687
	a-de	1068	0	0	933	272	672	6094		a-de	1511	0	0	638	409	375	4735
	a-ca	1113	0	0	697	177	500	2672		a-ca	1643	0	0	509	240	297	4125
	a-fr	1090	0	0	924	214	672	2906		a-fr	1635	0	0	638	310	390	5468
	a-uk	1172	3	0.25	921	235	672	3859		a-uk	1615	0	0	650	308	375	4297
	GW	1104	5	0.45	503	544	234	9375		GW	1363	0	0	1403	1544	265	9937
	GIP	1125	0	0	355	609	234	9360		GIP	1312	0	0	571	878	265	9594
hk	a-us	21002	81	0.38	448	304	250	9547	us	a-us	3725	0	0	74	135	31	3171
	a-jp	20944	11	0.05	388	321	203	9937		a-jp	3578	0	0	317	224	109	9219
	a-de	21130	729	3.45	573	346	343	9360		a-de	3766	0	0	298	271	109	9390
	a-ca	21255	125	0.58	440	286	250	9515		a-ca	3591	0	0	239	260	31	9515
	a-fr	21091	743	3.52	575	349	343	9703		a-fr	3933	0	0	433	222	187	3906
	a-uk	20830	807	3.87	570	348	328	9734		a-uk	3614	0	0	293	260	124	9157
	GW	21148	1426	6.74	1563	1560	406	9999		GW	3837	0	0	1290	1346	125	9828
	GIP	21007	1263	6.01	849	1582	203	9999		GIP	3621	0	0	675	1348	125	9938

- Timeout: 3865; Unavailable service (http 503): 2456; Bad gateway (http 502): 1
- Failure-rates are vary from location to location

5.1 Results-individual WS



•Response time performance (RTT) are vary from location to location.

5.2 Results-FT strategies

Table 4. Evaluation of Replication Strategies

Type	Cases			RTT(ms)			
	All	Fail	R%	Avg	Std	Min	Max
1	21556	6	0.027	279	153	203	3296
2	22719	0	0	389	333	203	17922
3	23040	0	0	374	299	203	8312
4	21926	4	0.018	311	278	203	10327
5	21926	1	0.004	312	209	203	10828
6	21737	2	0.009	311	225	203	10282
7	21737	2	0.009	310	240	203	13953
8	21735	0	0	411	1130	203	51687
9	21808	0	0	388	304	203	9360

- Strategy 1 (*Active*) has the best RTT performance, and the worst fail-rate.

- Sequential strategies (strategy 2 *Time*, strategy 3 *Passive*, 8: *Passive +Passive*, and 9: *Passive + Time*) obtain the worst RTT performance, and the best failure-rate.

5.3 Optimal strategy selection

Table 5. Scenario 1: Selection Procedure

$a = 20; b = 5; c = 5\%$

$n = 6; g = 21587; t_{user} = 100; f_{user} = 0.1\%$

$\{ws_i\}_{i=1}^{6}$ ={a-jp, a-us, a-ca, a-de, a-fr, a-uk};

$\{t_i\}_{i=1}^{6} = \{388, 448, 440, 573, 575, 570\};$

$\{f_i\}_{i=1}^{6} = \{0.05\%, 0.38\%, 0.58\%, 3.45\%, 3.52\%, 3.87\%\};$

$\{s_i\}_{i=1}^{6} = \{4.38, 8.28, 10.2, 40.23, 40.95, 44.4\};$

$\{\mathcal{T}(i)\}_{i=1}^{6} = (321, 285, 282, 281, 280, 279);$

$v = 1;$

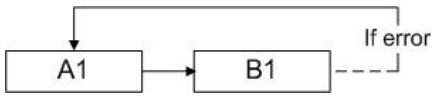
$W = \{ws_i | s_i =\leq 20 \&\& 1 \leq i \leq 6\} = \{4.38, 8.28, 10.2\};$

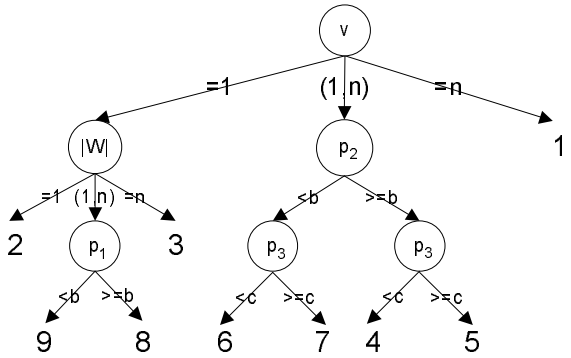
$|W| = 3;$

$p_1 = s_2 - s_1 = 3.9;$

$v = 1 \&\& 1 < |W| < 6 \&\& p_1 < 5 \Rightarrow$ Strategy 9;

Strategy 9: Passive+Time





$$s_i = \frac{t_i}{t_{user}} + \frac{f_i}{f_{user}}.$$

$$W = \{ws_i | s_i \leq a \&\& 1 \leq i \leq n\}$$

$$p_1 = s_2 - s_1$$

$$p_2 = \frac{1}{v} \sum_{i=1}^{v} (s_{i+v} - s_i)$$

$$p_3 = \frac{1}{v} \sum_{i=1}^{v} f_i$$

6. Conclusion and future work

- Conclusion
  - Distributed evaluation framework
  - Fault tolerance replication strategies.
  - Optimal replication strategy selection algorithm.
  - Experiment
    - More than 1,000,000 test cases.
    - Users from six locations.
    - Web Services located in six countries.
- Future work
  - Evaluation of stateful Web services.
  - Tuning of the selection algorithm
  - Investigating more QoS properties.

A Distributed Replication Strategy Evaluation and Selection Framework for Fault Tolerant Web Services

Zibin Zheng and Michael R. Lyu

Department of Computer Science & Engineering
The Chinese University of Hong Kong
Hong Kong, China

ICWS 2008, Beijing, China, 24 September, 2008