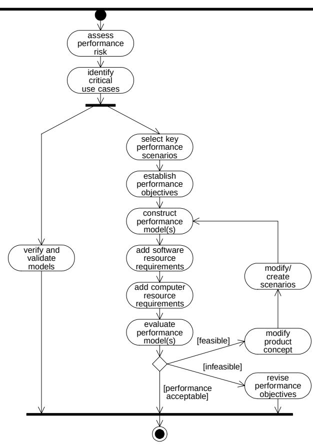
Research Topics

Objectives

- SPE Research Progress
  - Risk assessment
  - Workload selection
  - Performance requirements
  - Performance models
  - Resource requirements
  - Model solution technology
  - Evaluate results
  - Model V&V
  - Facilitating & motivating SPE
- Proactive modeling of designs: can the state be improved?

SPE Process



SPE Process Steps

1. Assess performance risk
2. Identify critical use cases
3. Select key performance scenarios
4. Establish performance objectives
5. Construct performance models
6. Determine software resource requirements
7. Add computer resource requirements
8. Evaluate the models
9. Verify and validate the models

1. Performance Risk Assessment

- Software Risk Assessment
  - Significant amount of work
  - Little of it addresses performance risks
- Need to quantify
  - Probability of performance failure
  - Severity of problem
- Build a business case for SPE (Smith & Williams CMG02 & 03)
- Companies are reluctant to publish failure data
- Need some way to build a data bank of this type of data

Cost/Benefit Worksheet

Cost/Benefit Worksheet			
One-Time Costs	**$**	**Cost Avoidance**	**$**
Tools		Refactoring	$ 812,500
Performance Modeling Tool	$ 8,000	Hardware Upgrade	$ 600,000
Load Driver	$ 70,000	Lost Revenue	$ 975,000
Workstation	$ 4,000	Telephone Agents	$ 325,000
Training			
In-House Training (15 Developers)	$ 66,846		
Performance Engineer	$ 5,923		
Consulting/Mentoring	$ 250,000		
Total One-Time Costs	**$ 404,769**	**Total Cost Avoidance**	**$ 2,712,500**
Recurring Costs (Annual)	**$**	**Intangible Benefits**	
Software Maintenance (Tools)	$ 12,100	Improved Corporate Image	
Salaries (Including Benefits)		Enhanced Customer Relations	
Performance Analyst (1.0 FTE)	$ 100,000	Improved Employee Morale	
Continuing Education	$ 2,200		
Total Recurring Costs	**$ 114,300**		

Return on Investment

Cost Benefit Summary		
Number of Years		1.5
	Cost	Benefit
One-Time	$ 404,769	$ 2,712,500
Recurring	$ 171,450	
Total	$ 576,219	$ 2,712,500

$$ROI = \frac{\$2,712,500}{\$576,219} = 417\%$$

2-3. Workload Selection

- Identify critical use cases & select performance scenarios
- Early work characterized workloads for system execution models from measurements (Ferrari 72, Haring 82)
  - SPE can use these techniques for evolutionary development and replacement systems
  - New systems must forecast intensity and importance of use cases
- Expert system for developers to select performance scenarios?

4. Performance Requirements

- Have notations for specifying requirements, but little formal basis for determining what they should be
  - Reactive systems often have well-defined performance requirements
  - Human user interactions are more difficult to specify, especially end-to-end work tasks
- Promising Approach: APDEX Application Performance Index www.apdex.org
  - Measure, rate and report application performance
  - Good basis for establishing performance requirements
- Vital for SPE – much work is needed:
  - Realistic, complete, consistent?
  - Testable?
  - Automatic construction of performance tests
  - Automation for V&V?

5. Performance Models

- Well-developed area
  - Execution graphs & supporting tools
  - Executable code
  - State machines
  - Petri nets & stochastic process algebra
  - Component-based systems & supporting tools
- Automatic translation of UML to performance models - easier for developers to create models from specifications
- Developers need tools that
  - Identify critical parts of software
  - Focus on simple models of those parts
  - Fill in details for those parts
  - Exclude unimportant details

6-7. Resource Requirements

- Software resource requirements – meaningful from software perspective
  - # messages sent, # database accesses, etc.
  - Easier for developers to estimate
- Computer resource requirements – path lengths for software resources
  - CPU time, disk I/Os etc.
  - Usually from measurements
- Vital to provide assistance for developer estimation, but difficult to plan research that will overcome difficulties
  - Develop some reasonable default bounds?
  - Develop parameters for re-used components?
  - Make it easier to get measured values?

8. Model Solution Technology

- Early work on approximate analytic solutions for complex systems
- Today's processing speed makes simulation viable
  - Still need to screen models before simulating
- Still need simple, fast solution techniques for complex systems
  - Quickly and easily find serious architecture or design problems
- Need
  - Further de-skill modeling task for technology transfer for developers – they won't use the complex, one-of-a-kind models that require extensive background to use
  - Take research results a few steps further

Extensions to Model Technology

- Examples
  - Web -> heavy tailed distributions
  - Web Services -> distributed system models
  - Service Oriented Architecture
- Needed before the technology is widely deployed
- We're doing a good job in this area!
  - Ref - Smith & Woodside 99, System Performance Evaluation: Methodologies and Applications, CRC Press

Evaluate Results

- Performance antipatterns characterize common (software) architecture and design problems and how to correct them (WOSP, CMG)
  - E.g., Excessive dynamic allocation, One lane bridge, etc.
- Need:
  - Automatic detection
  - Suggested solutions
  - Quantify all costs of various solutions

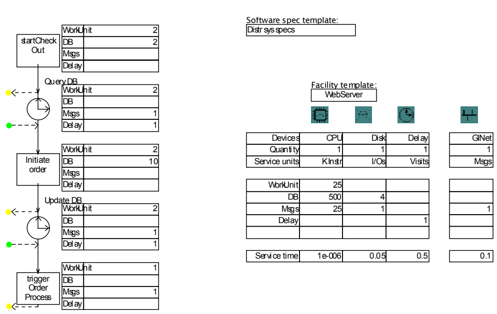
9. Model Verification and Validation

- Measurements
  - System level and fine-grained software data
  - Compare to models
  - Predictions
- Need more automation of these tasks

Measurements for SPE

- Problems
  - SW often developed on different platform than deployed
  - Test data seldom representative of performance workloads
    - Volume not representative
    - Content seldom reflects key performance scenarios
- Difficult to use measurement tools and extract data
  - Most tools are intended for performance tuning

SPE Data Needed



Difficulty Getting Data

```
** CPU USAGE BY SQL STATEMENT **

 1105 SELECT                          .01   .01  .
 1137 SELECT                          .04   .04  .
 1153 SELECT                          .01   .02  .
 1169 SELECT                          .03   .04  .
 1200 SELECT                          .07   .15  .
 1224 SELECT                          .05   .06  .
 1239 SELECT                          .05   .09  .
 1255 SELECT                          .08   .09  .
 1270 SELECT                          .04   .05  .
 1286 SELECT                          .02   .03  .
  531 INSERT                        20.57 29.41  .**************************++++++++++++
  565 INSERT                         6.00  8.50  .********+++
  760 INSERT                         3.03  4.30  .****+
  651 UPDATE                          .32   .42  .
  591 OPEN        01 DECLARE              3.36  5.09  .****++
  606 FETCH       01 DECLARE               .02   .02  .
  698 FETCH       01 DECLARE               .06   .10  .
                              -----  -----
DBRM - SQL94158          TOTALS         33.98  48.70
```

Processing steps? Application code? I/Os? Amount of data returned? System overhead?

Measurements for SPE Data

- All modeling tools need similar data
  - it would be nice to agree on some standard XML tags and have measurement tools export data for lots of modeling and analysis tools to use
- Our PMIF and SPE meta model are a starting point – they define info requirements for SPE [Tools 95, JSS 99]
- Remember the data for validation as well as model parameterization

SPE Motivation: First Project

- Constructed SPE models during design
- Models predicted best case response time >1 hr.
- Project did not implement recommended changes due to schedule constraints
- Subsequent performance problems made integration testing impossible
- Many problems were due to (predicted) design problems, tuning alone could not correct them
- $20 Million Project canceled
- → **Model technology is not the problem**

Technology Transfer

- Needs to be much quicker and easier to do model studies
- Make it accessible to system developers rather than modeling gurus
- Build tasks and deliverables into development process
- Package solutions such as PASA
- Standardized solutions such as UML

Education: Practitioner's Perspective

- If you want developers to use SPE, teach them from the beginning that it is the correct way to build software!
  - Old dogs and new tricks?
- < 50% of developers have degrees with CS major or minor. Even those that do may not take performance modeling courses.
  - Some challenges

Conclusions

- We've made lots of progress, the field is maturing
- Modeling research
  - Simplification is valuable
  - Model parameterization and validation is too hard
  - New technology will always provide interesting problems to solve
- State of the Practice
  - Need packaged solutions tailored to problems
  - Need known, standardized, accepted solutions
  - Need to make it easier for practitioners to use
  - Need to integrate into development process – treat performance as a functional requirement
- Education and Research
  - Basic performance knowledge is essential
  - Need an effective way to communicate results to those who need them

Summary

- SPE Research Progress
  - Risk assessment
  - Workload selection
  - Performance requirements
  - Performance models
  - Resource requirements
  - Model solution technology
  - Evaluate results
  - Model V&V
  - Facilitating & motivating SPE
- Proactive modeling of designs: can the state be improved?
- Education: from a practitioners perspective

