Software, Performance, or Engineering?

Daniel A. Menascé
Department of Computer Science
George Mason University
Fairfax, VA, USA
www.cs.gmu.edu/faculty/menasce.html

Some Definitions

(according to CMU's SEI)

- Engineering:
  - Systematic application of scientific knowledge in creating and building cost-effective solutions to practical problems in the service of mankind
- Software Engineering:
  - Form of engineering that applies principles of computer science and mathematics to achieving cost-effective solutions to software problems.

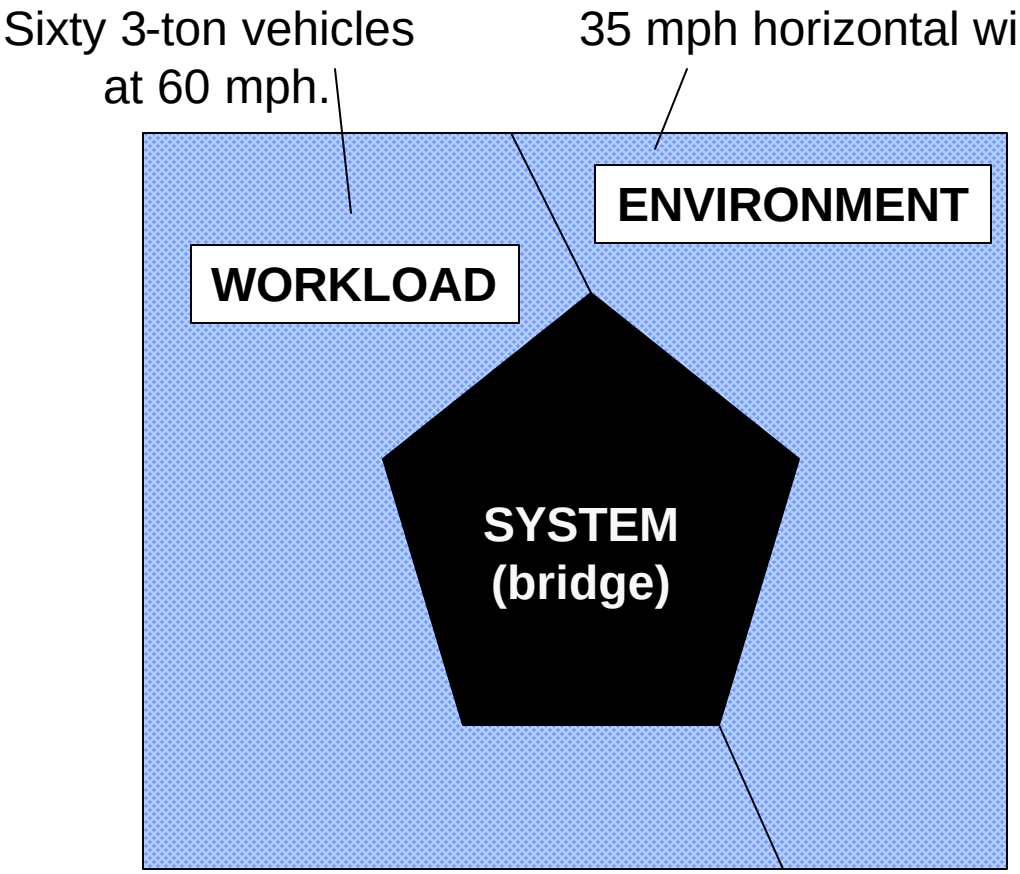
What is (or should be) SE?

Systematic application of principles of computer science and mathematics to the design, maintenance, and evolution of software systems in such a way that all of its requirements---functional and non-functional---are met.

A bit of history …

- Software Engineering (SE): around for more than 35 years (1968 NATO conference)
  - Success in developing methods for programming in the small and in the large.
  - Methods for taming the complexity of software development
  - Methods and tools to develop and manage designs, requirements, test cases, configurations, versions, and evolution.

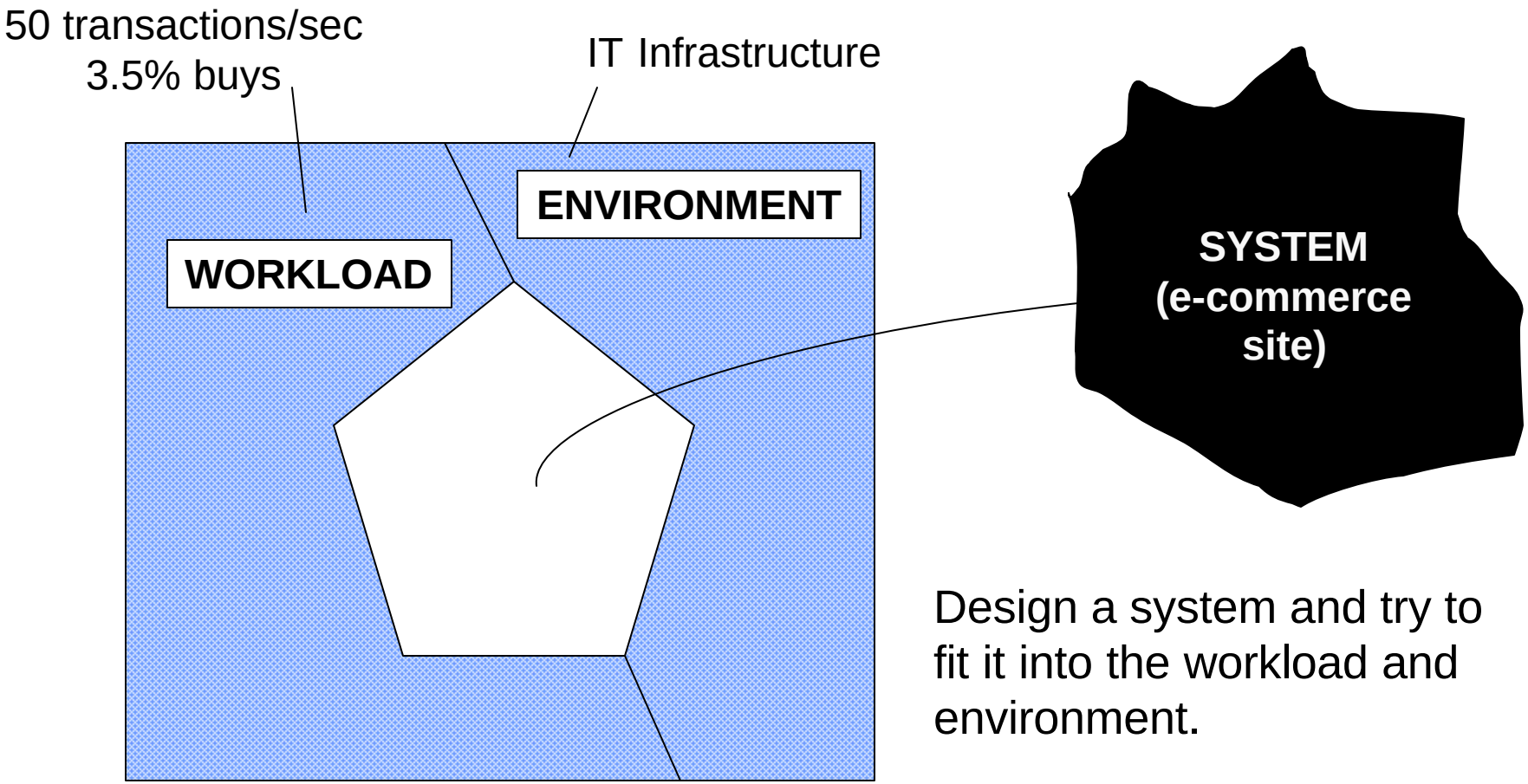
Conventional Engineering



Systems are designed with the workload and environment in mind

Perfect fit.

Software Engineering



Design a system and try to fit it into the workload and environment.

Does not always fit!

What is (or should be) SE?

Systematic application of principles of computer science and mathematics to the design, maintenance, and evolution of software systems in such a way that all of its requirements---functional and non-functional---are met.

Why performance requirements are called non-functional?
How can a software system work properly if some of its requirements (e.g., performance requirements) are not met?

All requirements should be called functional!
Otherwise, they should not be requirements.

More history …

- 21 years ago Connie Smith coined the term Software Performance Engineering (SPE) (CMG'81 paper).
- Claims:
  - "fix-it" later attitude when it came to performance.
  - performance was never a design consideration.

Some questions …

- Is the term “Performance” in “Software Performance Engineering” redundant?
  - If SE is Engineering, then it should produce efficient systems by definition.
- Does it make sense to talk about “Efficient Mechanical Engineering?”
  - No. Mechanical engineers (and all other CEs) strive to design efficient mechanisms.

The reality is …

- Performance in SPE is not yet redundant!
- The SPE concepts introduced by Connie Smith over 20 years ago have not yet been incorporated into mainstream software engineering.

Where is the P in SE?

1. Lack of scientific principles and models.
   - CEs need to rely on principles and models from mathematics, physics, and computational sciences to design their systems.
   - SEs do not need to rely on formal and quantitative models to design and build their systems.

Where is the P in SE?

1. Lack of scientific principles and models.
   - SE: many advances in terms of formal models to support the software development life cycle: development, maintenance, and evolution. No universally agreed upon formalisms and quantitative models to support the core of SE.
   - SE: Most of the energy devoted to formalisms and methods that support “functional” requirements (over 80% of IEEE TSE papers since 1989).

Where is the P in SE?

1. Lack of scientific principles and models.
   - PE: mostly addresses issues of system performance from a resource demand point of view.
   - PE: most models do not model the application directly, only the demands it places on the physical resources (e.g., QN models).
   - PE: layered QN-type models represent the application explicitly. These models are not as widely-known and -adopted as conventional QN models.

Where is the P in SE?

2. Education and Curricula Problems.

- The vast majority of undergraduate CS and related curricula do not include any required course in performance evaluation. Exceptions: minimal performance-related hours in OS and networking courses.
- Joint IEEE CS/ACM Task Force on the "Model Curricula for Computing" draft CS curriculum: performance is overlooked!

Joint IEEE CS/ACM Task Force on the "Model Curricula for Computing"

- CS divided into 14 areas:
  1. Discrete Structures (DS)
  2. Programming Fundamentals (PF)
  3. Algorithms and Complexity (AL)
  4. Programming Languages (PL)
  5. Architecture and Organization (AR)
  6. Operating Systems (OS)
  7. Net-Centric Computing (NC)
  8. Human-Computer Interaction (HC)
  9. Graphics and Visual Computing (GV)
  10. Intelligent Systems (IS)
  11. Information Management (IM)
  12. **Software Engineering (SE)**
  13. Social and Professional Issues (SP)
  14. Computational Science and Numerical Methods (CN)

Joint IEEE CS/ACM Task Force on the "Model Curricula for Computing"

- SE is divided into 12 areas:
  - **SE1. Software design [core]**
  - **SE2. Using APIs [core]**
  - **SE3. Software tools and environments [core]**
  - **SE4. Software processes [core]**
  - **SE5. Software requirements and specifications [core]**
  - **SE6. Software validation [core]**
  - **SE7. Software evolution [core]**
  - **SE8. Software project management [core]**
  - **SE9. Component-based computing [elective]**
  - **SE10. Formal methods [elective]**
  - **SE11. Software reliability [elective]**
  - **SE12. Specialized systems development [elective]**

Where is the P in SE?

2. Education and Curricula Problems.

- Why isn't computer performance part of the curriculum?
  1. Lack of training in performance by faculty who teach SE.
  2. Lack of universally agreed upon methods and models to be used by SEs to address performance issues.
  3. Faculty resistance to change.
  4. Limits on total number of hours in the curriculum.

Where is the P in SE?

2. Education and Curricula Problems.

- How do students view the importance of software issues in the performance of computer systems?
  - 5-question test answered by 59 grad and undergrad students:
    - ✓ 19 seniors in the BS in CS program.
    - ✓ 39 graduates in MS in CS and MS in SE programs.

Test Questions

1. Define response time (RTDef)
2. What units are used to indicate response time? (RTU)
3. Define throughput. (XDef)
4. What units are used to indicate throughput? (XU)
5. Identify possible factors that contribute to the time taken by a Web search engine to return a reply to a browser. (RTFactors)

Level	No. Students	% took system courses	% took SE courses	Percent Answered Correctly					
				RTDef	RTU	Xdef	XU	RTFactors	Software Factors
Senior in BS	19	100	84	100	74	58	63	84	21
Graduate	39	95	44	87	82	74	54	87	5

Most students did well on basic performance questions.

A very high percentage of students was able to correctly identify factors that impact response time.

A very low percentage included software issues as factors that impact response time.

Where is the P in SE?

3. IT Workforce Issues.

- 2.5 million core IT workers (computer engineers, computer system analysts and scientists, programmers, and computer science teachers) in 1999 in the US (source: National Research Council).
- IT workforce needs to grow by 175,000 a year (source: US Bureau of Labor and Statistics).
- 42,000 bachelor degrees in CS and engineering awarded in 2000 in the US and Canada.

Where is the P in SE?

3. IT Workforce Issues.

- Computer scientists, computer engineers, system analysts, and programmers in 1998*:
  - 67% held a BS or higher degree.
  - 33% (mostly programmers) had a 2-yr college degree or HS diploma.
  - < 50% had a bachelors or other degree with a major or minor in CS or CS-related discipline!

Many individuals without formal training are employed in IT and learn on the job!

* source: US BLS

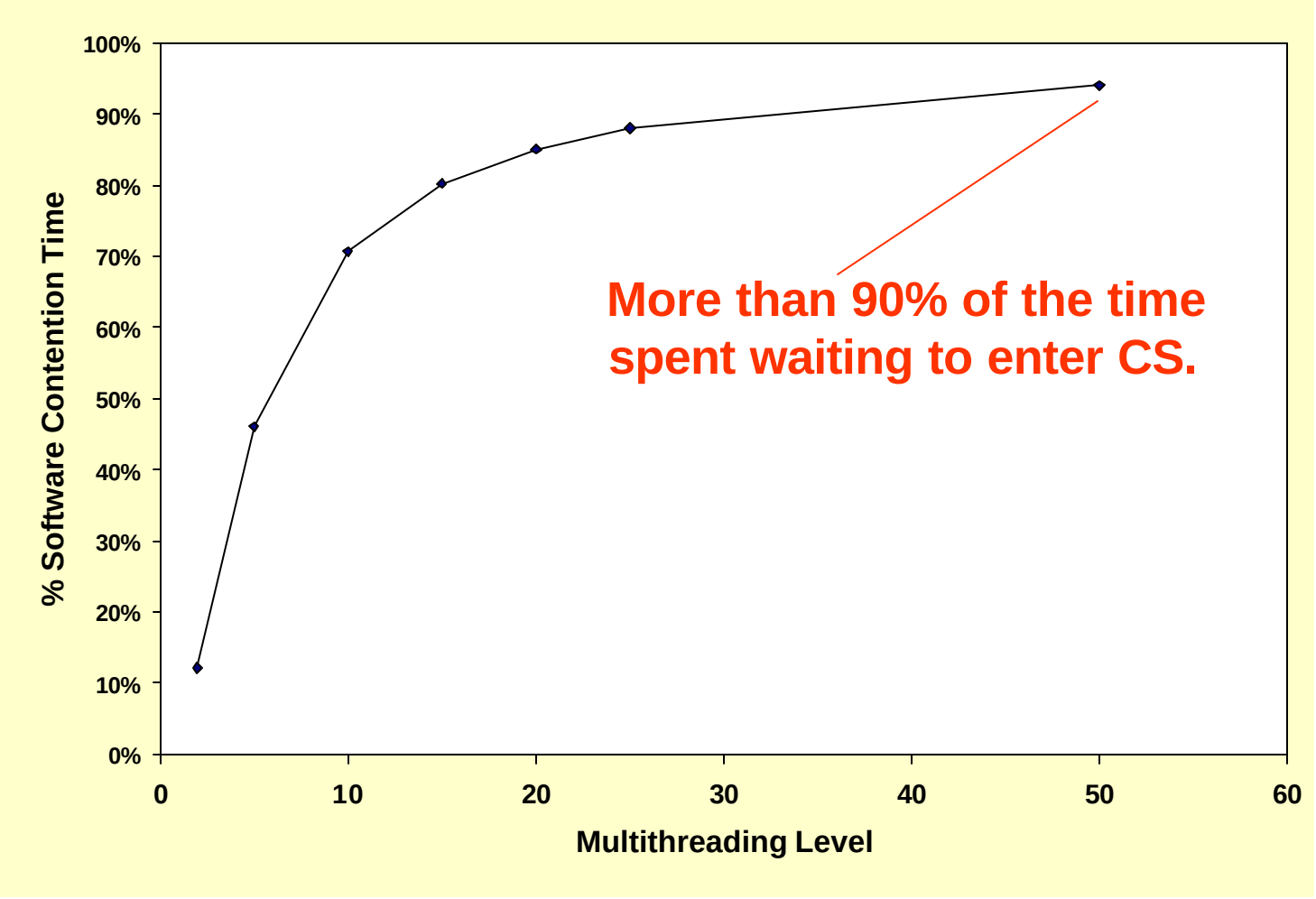
Where is the P in SE?

4. Single-user mindset.

- Concurrency overlooked by people who write code.
  - Concurrency for physical resources.
  - Concurrency for software resources (database locks, critical sections, and software threads)

Where is the P in SE?

4. Single-user mindset.



Where is the P in SE?

5. Small database mindset.

- Most programmers write database access code without taking into account the size of the database.
- Most tests are done in small development databases used to test functionality.
- The performance of an SQL call on a DB with 1,000 rows may be significantly different than that on a DB with one million rows.

Where is the P in SE?

1. Lack of scientific principles and models.
2. Education and curricula issues.
3. IT workforce problems.
4. Single-user mindset.
5. Small database mindset.

What Can We Do?

- In education and training
  - Seamless integration of performance concepts into software engineering courses and degree programs at all levels.
- In research
  - Development of universal models and methods that can be easily used by software developers.
  - Development of QoS-aware software architectures and components.
- In industry
  - Development and adoption of integrated tools that facilitate integration of performance management and instrumentation in the software development life cycle.

More on Research Issues

- At software design time:
  - Seamless integration of performance modeling with architecture and software design methods (e.g., performance annotation of UML designs)
    - OMG's Response to the OMG RFP for Schedulability, Performance, and Time, June'01.
    - OMG's RFP: UML Profile for Modeling QoS and Fault Tolerance Characteristics and Mechanisms, February'02.
  - Automatic generation of performance models from design specifications.

More on Research Issues

- At software design time:
  - Seamless integration of performance modeling with architecture and software design methods (e.g., performance annotation of UML designs)
    - OMG's Response to the OMG RFP for Schedulability, Performance, and Time, June'01.
    - OMG's RFP: UML Profile for Modeling QoS and Fault Tolerance Characteristics and Mechanisms, February'02.

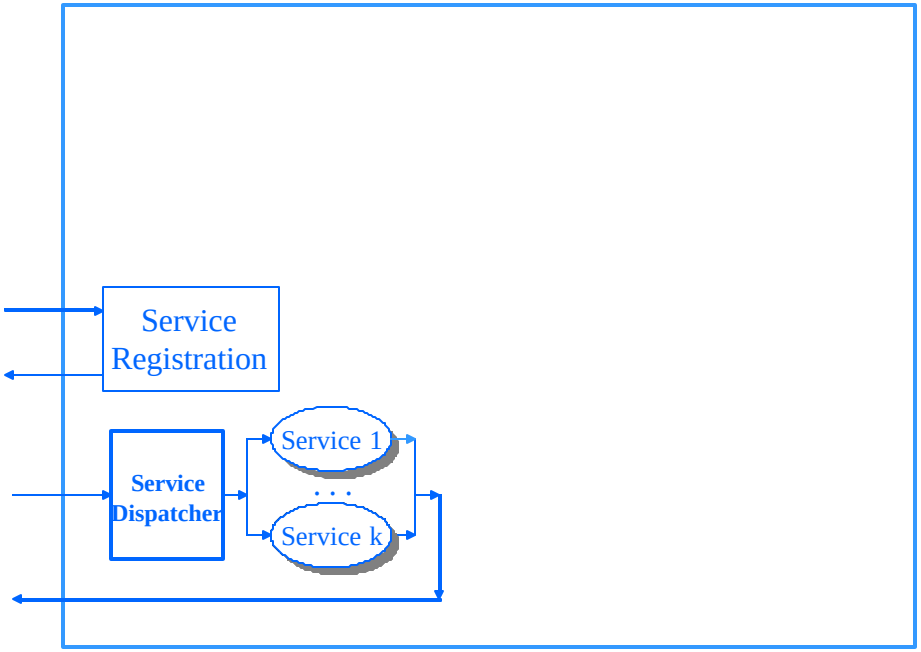
How general? How easy to use? Is it universal?

How do you get the parameters?

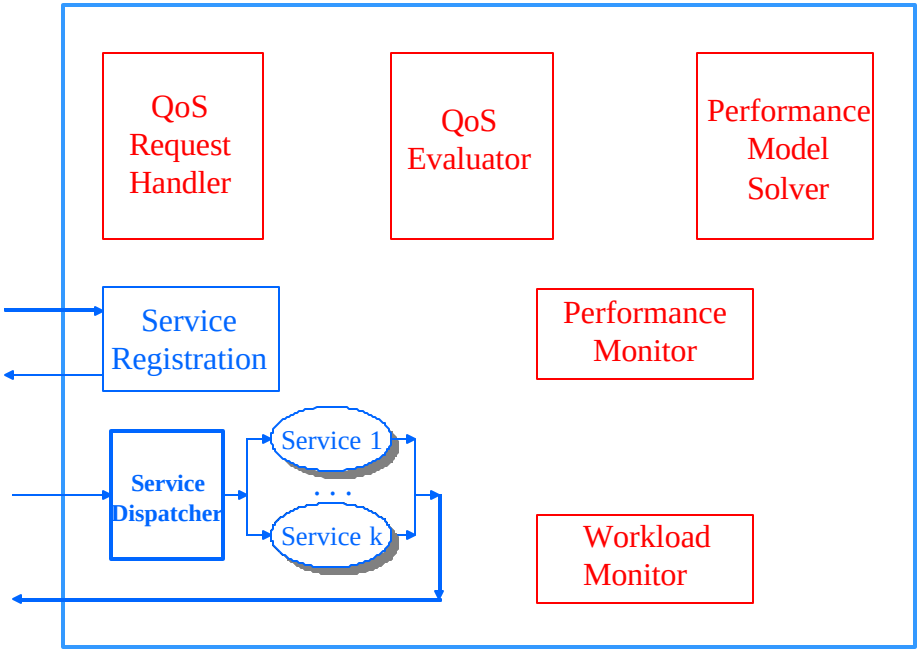
More on Research Issues

- At run-time:
  - QoS-aware software components: local and global configuration and admission control.

Architecture of a component



Architecture of a QoS-aware component



More on Research Issues

- At run-time:
  - QoS-aware software components: local and global configuration and admission control.
  - Use of performance models to build dynamically controlled and self configurable software systems.
    - Example: Dynamically controlled E-commerce site (see next few slides), ACM E-commerce Conference, 2001, Menascé, Barbara, and Dodge.

Dynamic QoS Control for E-commerce

- Definition of a combined QoS metric.

QoS Deviation

- Relative difference between the observed QoS value and the QoS goal.

Response time deviation:

$$\Delta QoS_R = \frac{R_{\max} - R_{measured}}{R_{\max}}$$

Throughput deviation:

$$\Delta QoS_X = \frac{X_{measured} - X_{\min}}{X_{\min}}$$

Probability of rejection deviation:

$$\Delta QoS_{P_{rej}} = \frac{P_{rej}^{MAX} - P_{rej}^{measured}}{P_{rej}^{MAX}}$$

A negative deviation means that the QoS level for the metric has not been met.

QoS Metric

- The QoS metric is defined as a linear combination of QoS deviations.
- The weights are determined by management based on the relative importance of each metric.

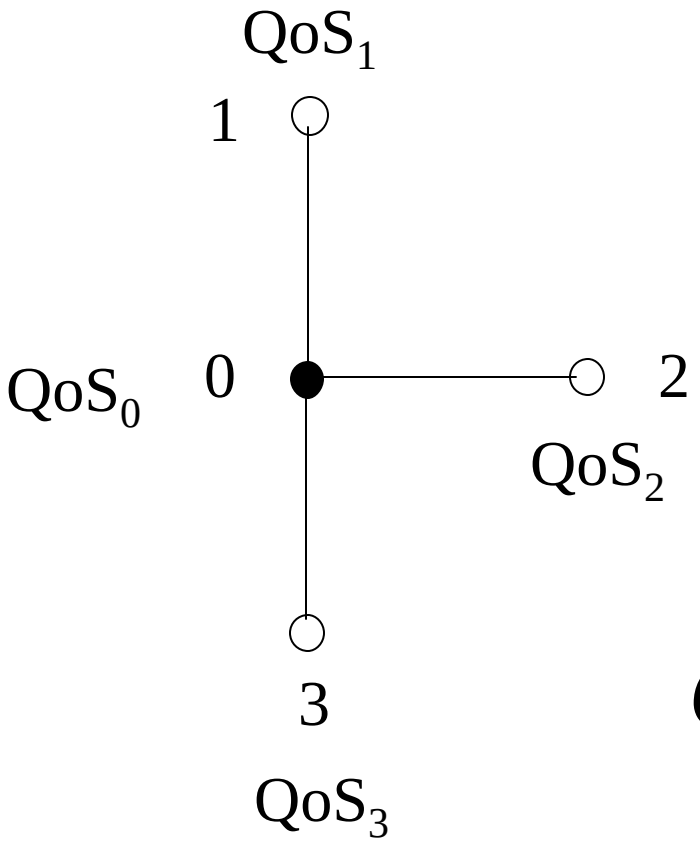
$$QoS_v = (\Delta QoS_R \times W_R) + (\Delta QoS_{\text{Prej}} \times W_{\text{Prej}}) + (\Delta QoS_X \times W_X)$$

determined by management

Dynamic QoS Control for E-commerce

- Definition of a combined QoS metric.
- Use of hill-climbing techniques combined with predictive queuing models.

Heuristic Optimization Approach

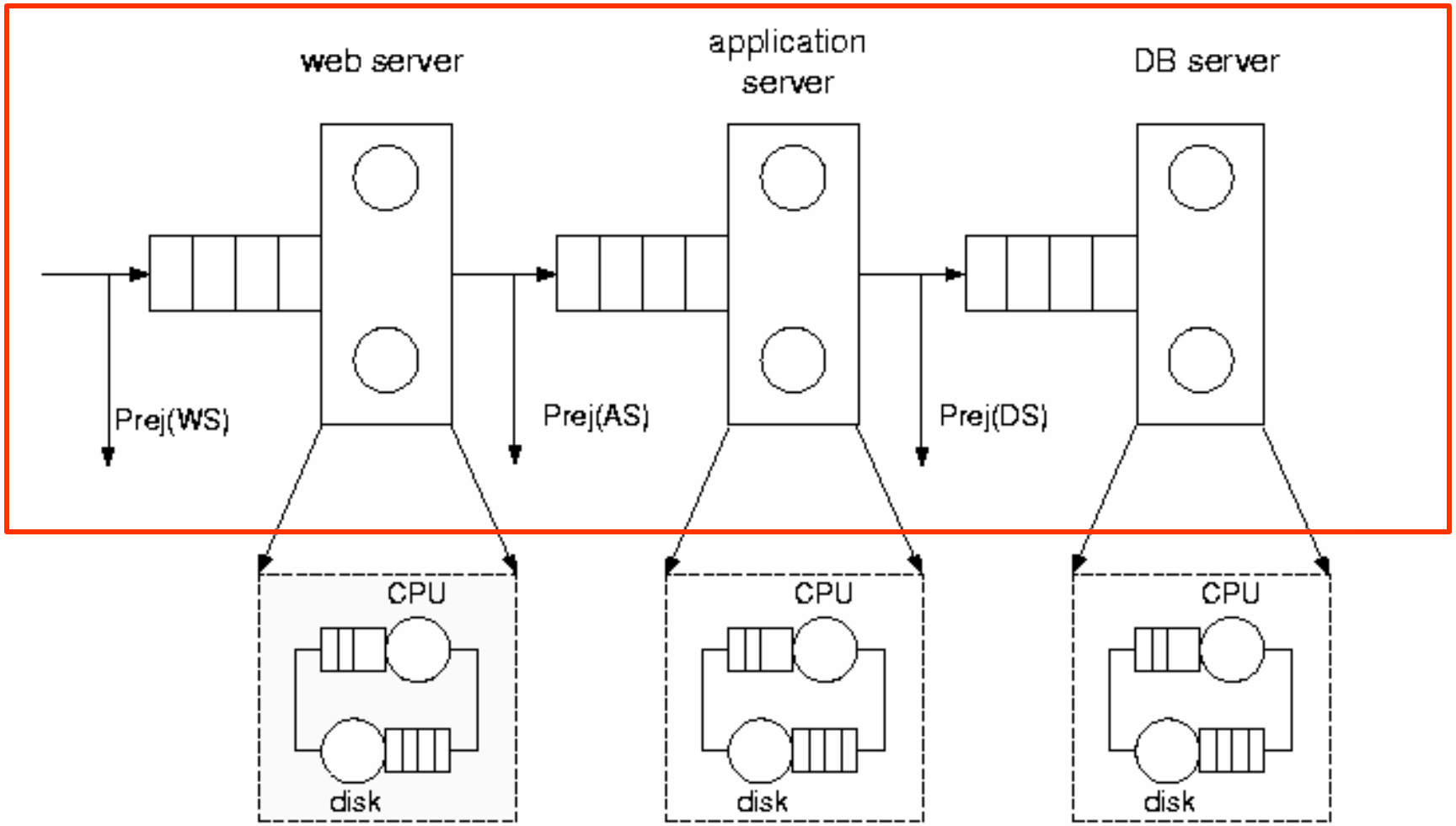


QoS_0: observed QoS value for current configuration

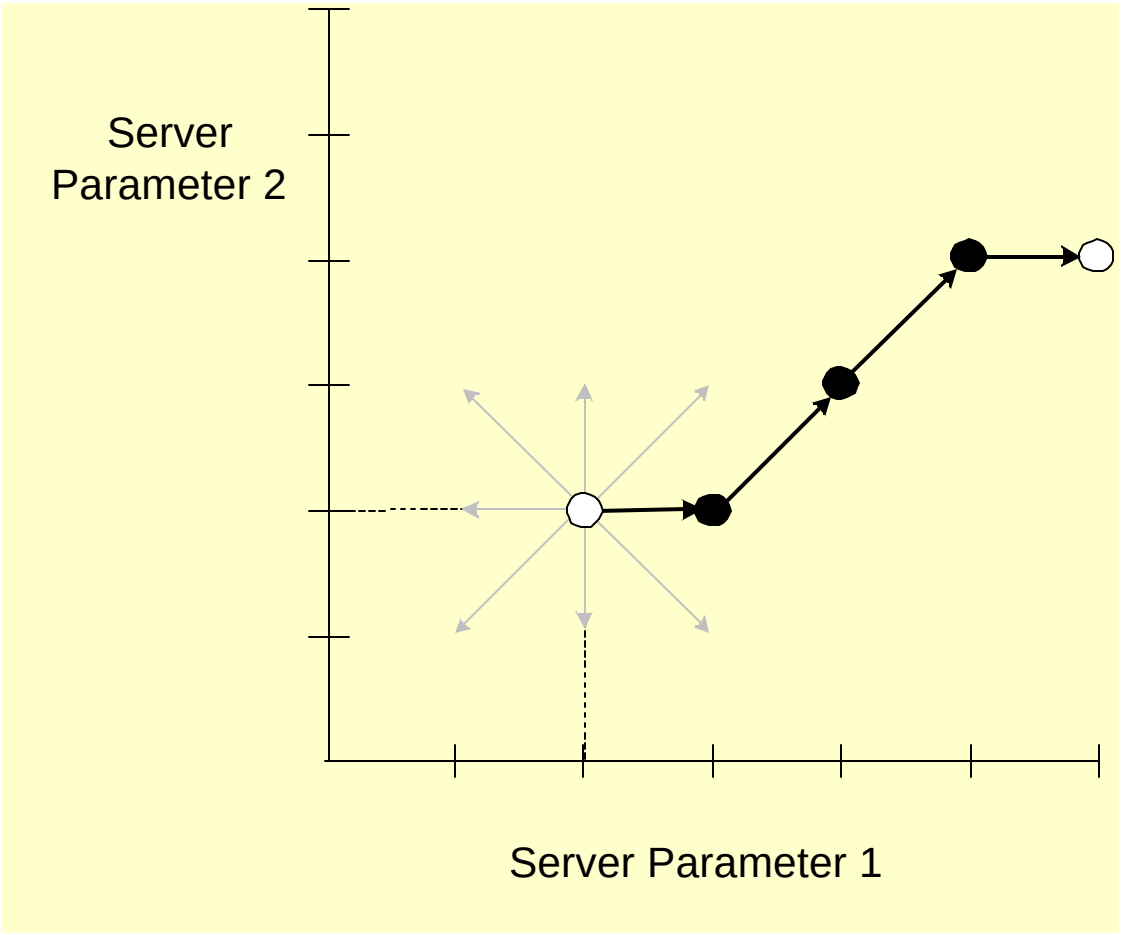
QoS_1, QoS_2, and QoS_3 are determined by a predictive queuing model of the site.

$$QoS = f(\vec{W}, c_1, c_2, ..., c_m)$$

E-commerce Site Queuing Model



Heuristic Optimization Approach

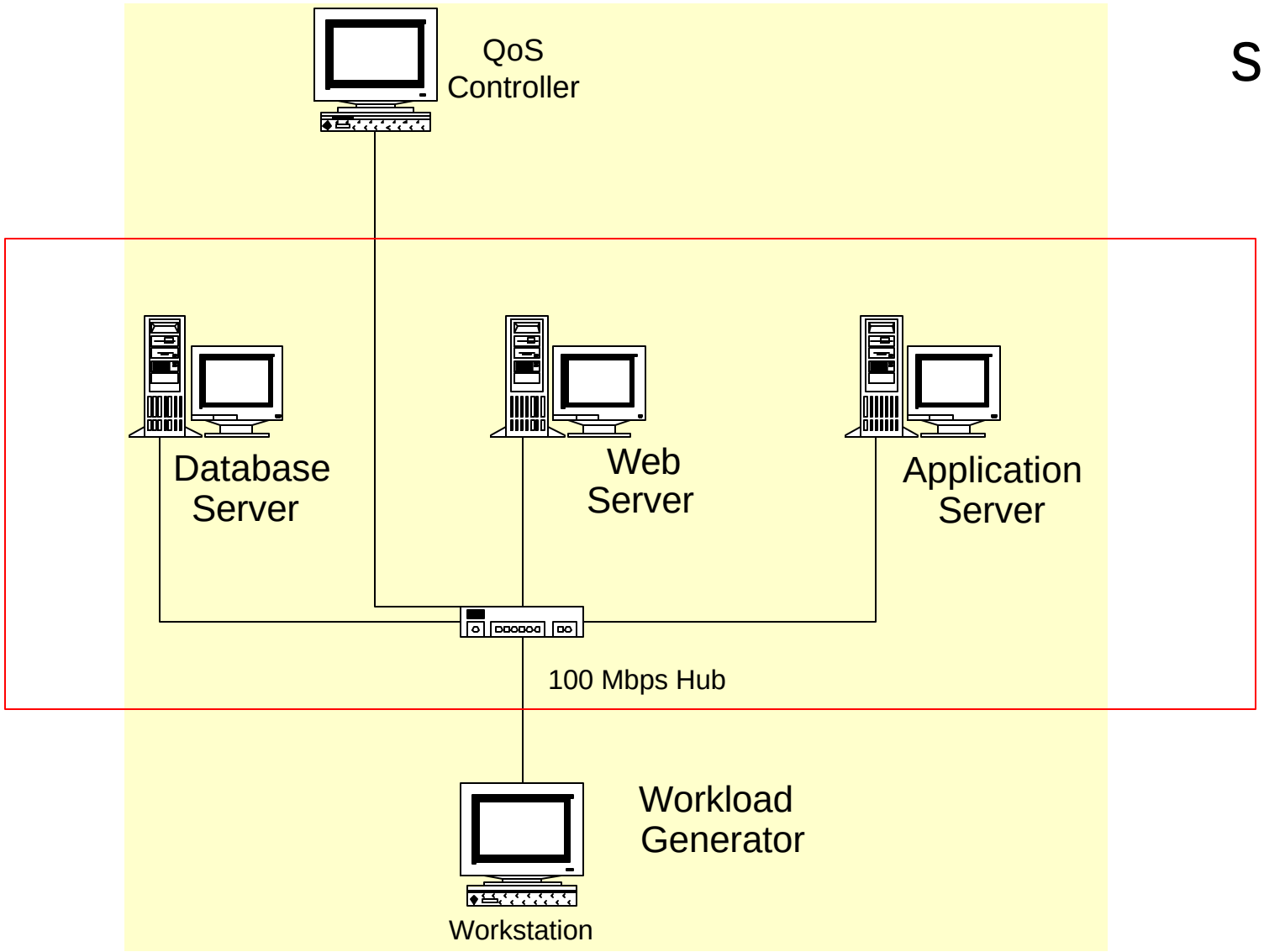


- Parameters for the queuing model are collected dynamically from the site.
- The QoS values for surrounding points are calculated
- The path with the greatest QoS gain is chosen. If no configuration improves the QoS or a limit is reached, the search ends

Dynamic QoS Control for E-commerce

- Definition of a combined QoS metric.
- Use of hill-climbing techniques combined with predictive queuing models.
- Implemented a TPC-W site in a multi-tier architecture.

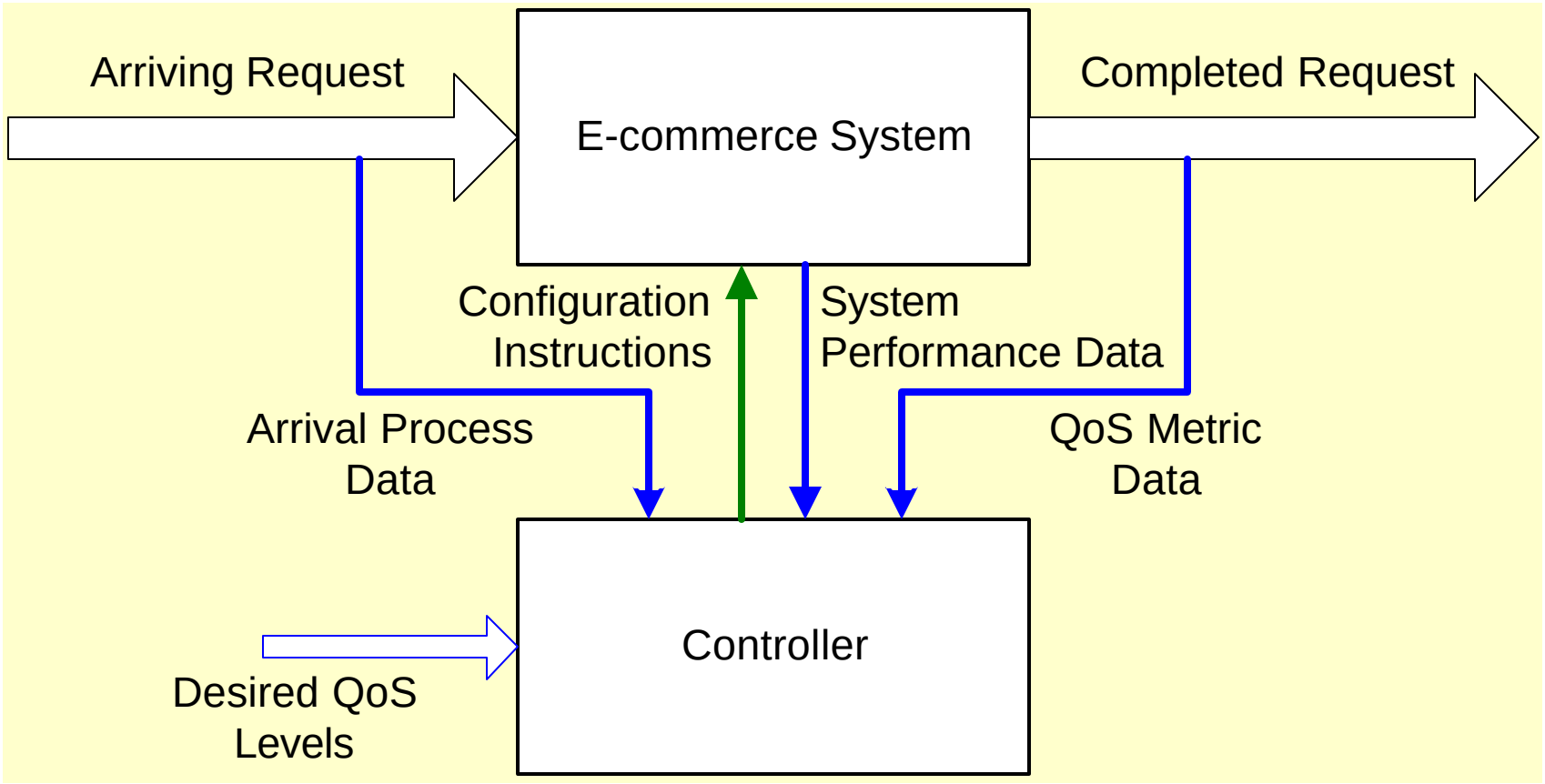
Prototype Configuration



Dynamic QoS Control for E-commerce

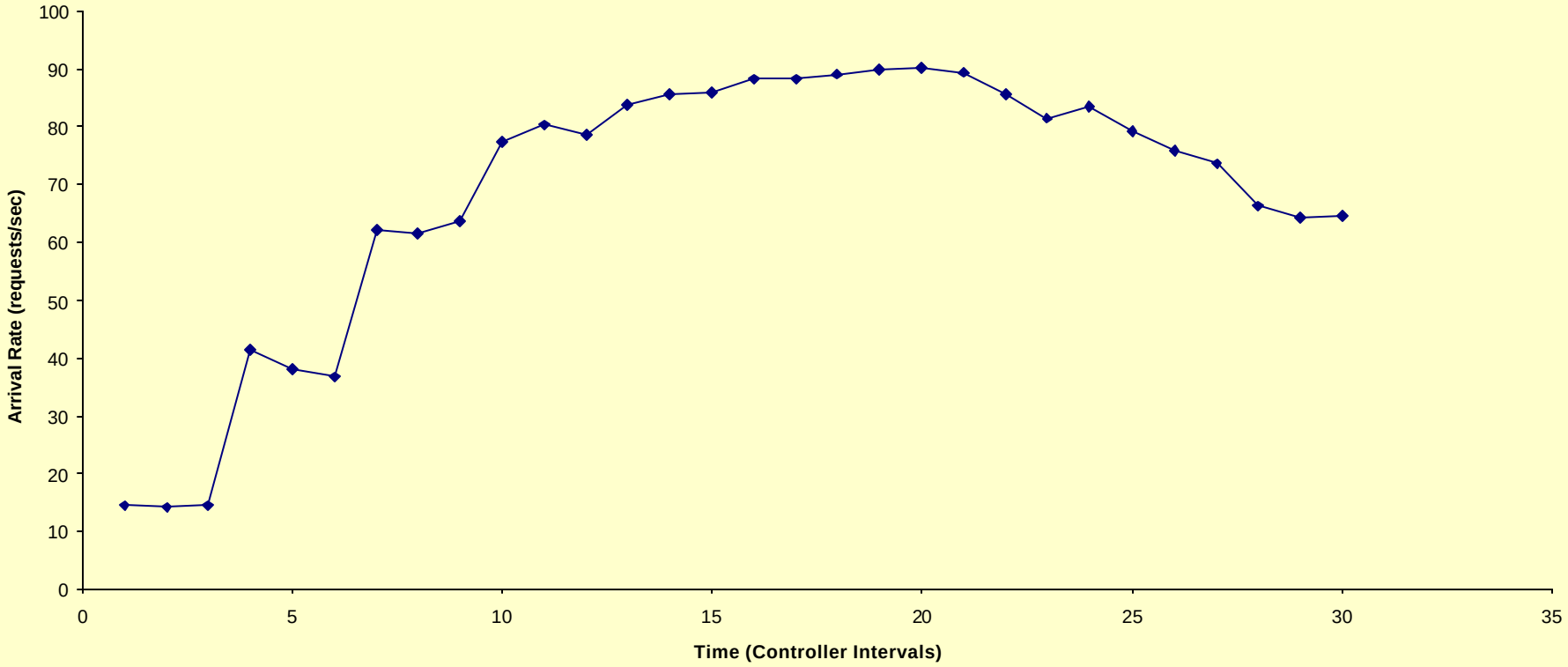
- Definition of a combined QoS metric.
- Use of hill-climbing techniques combined with predictive queuing models.
- Implemented a TPC-W site in a multi-tier architecture.
- Implemented a QoS Controller on a dedicated machine and evaluated the approach.

Dynamic QoS Controller

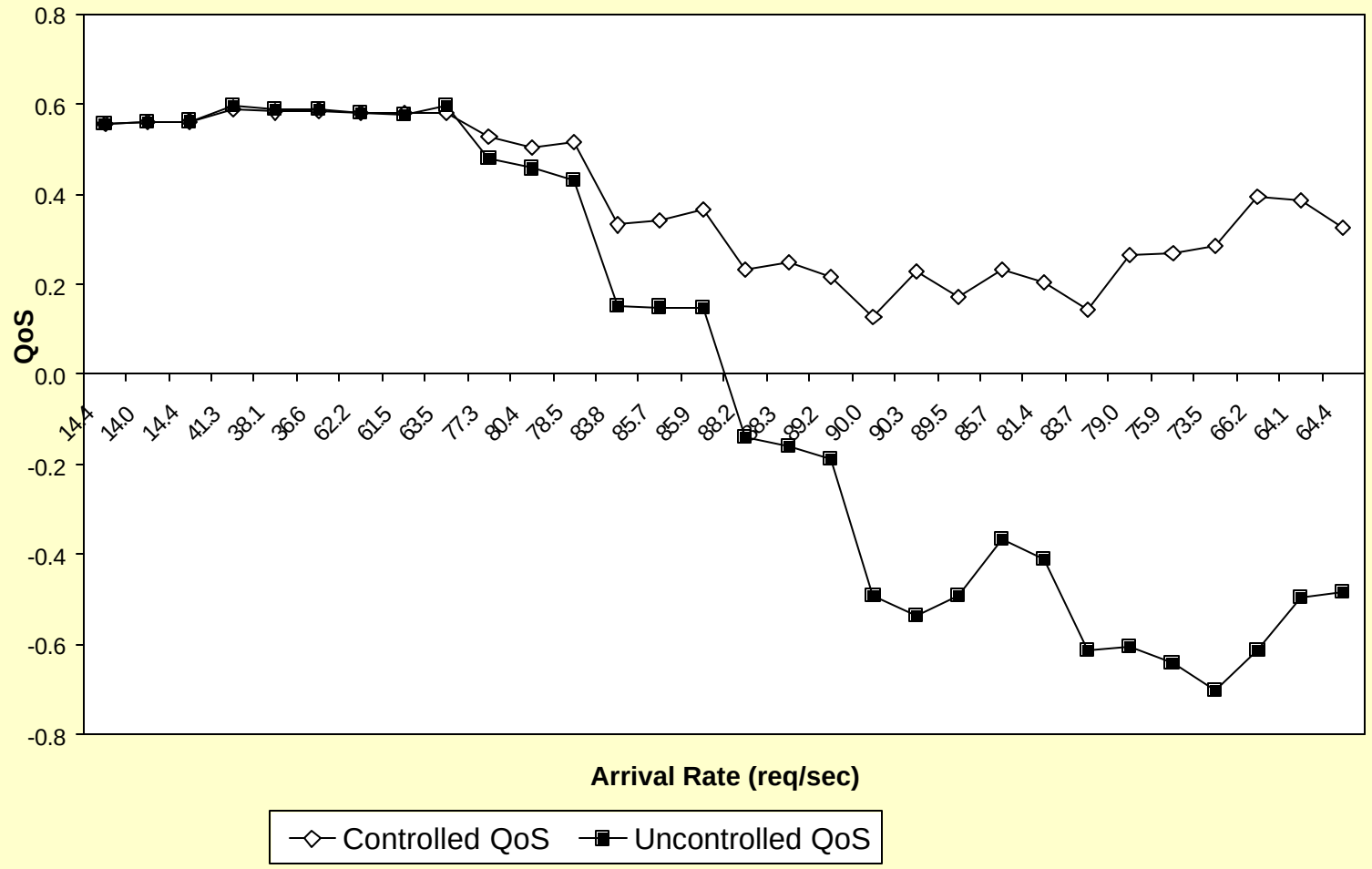


Experiment Results

Arrival rate

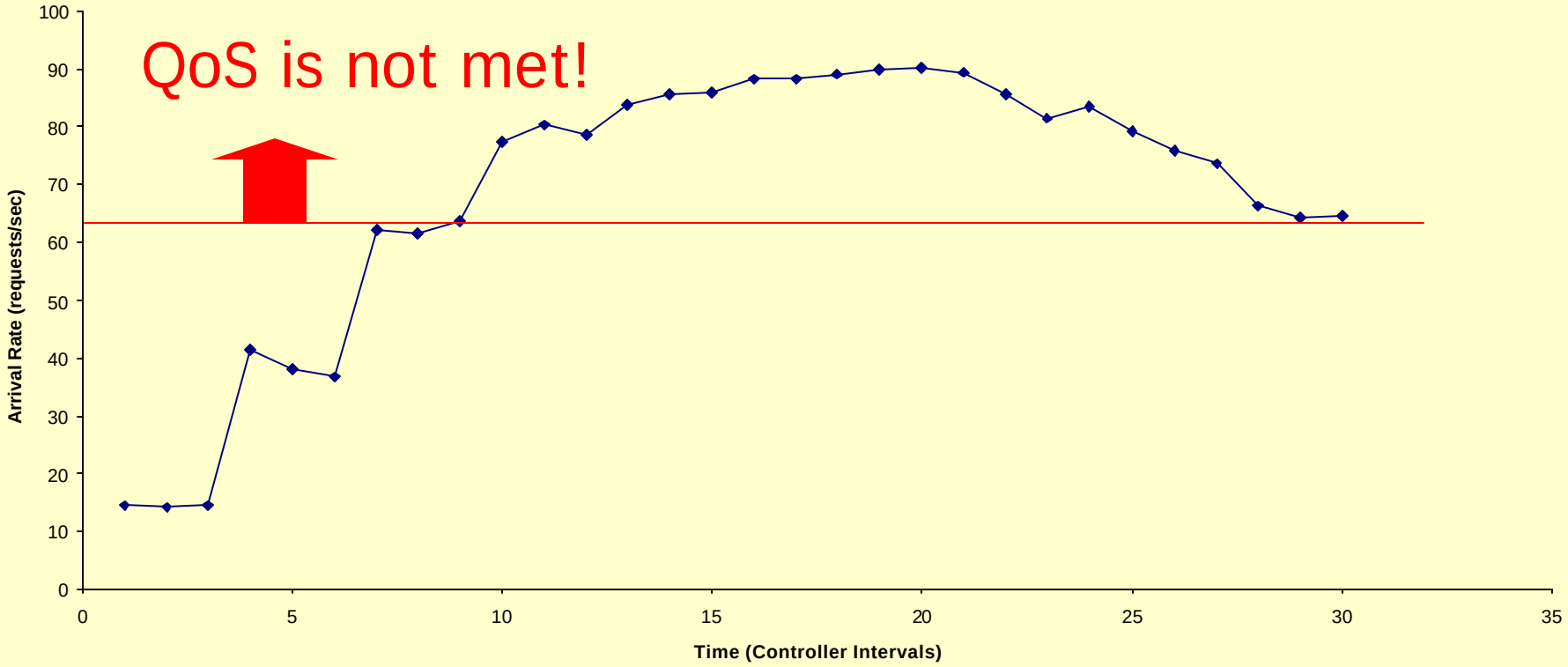


Results of QoS Controller



Experiment Results

Arrival rate



Future Directions

- At run-time:
  - QoS-aware software components: local and global configuration and admission control.
  - Dynamically controlled and reconfigurable software systems.

Must deal with components discovered at run time (see Web services).

Our goal should be …

SPE

Our goal should be …

SE

… to make SE incorporate SPE by definition.

slides will be at
www.cs.gmu.edu/~menasce/papers/wosp02-keynote.pdf

Questions?