Some Software Performance Challenges in 2016

A Presentation to WOSP-C 2016

Workshop on Challenges in Software Performance Engineering

A Workshop of ICPE 2016, Delft, Mar 12 2016.

Murray Woodside

Real-time and Distributed Systems Lab,

Carleton University, Ottawa, Canada

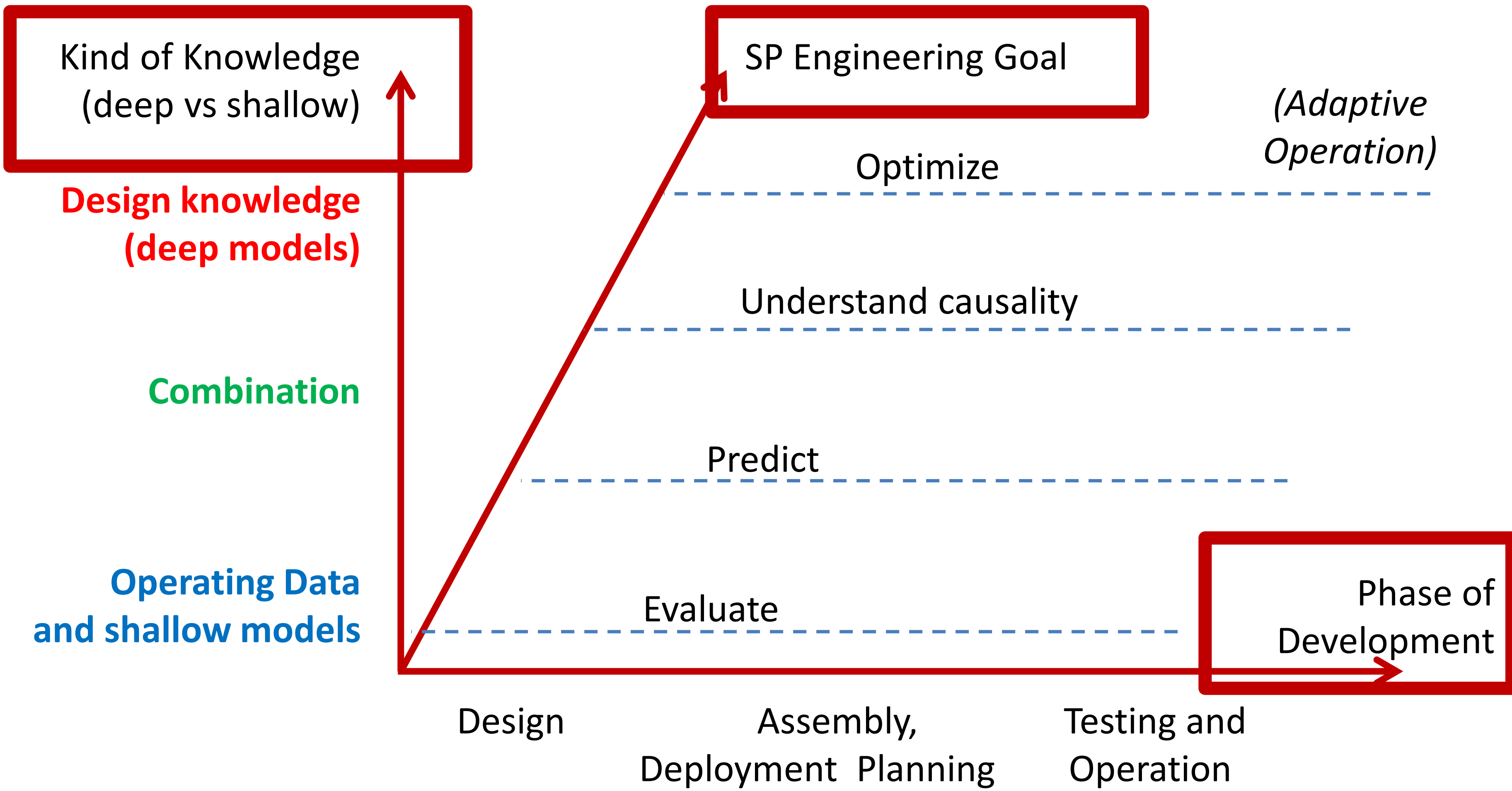
Carleton UNIVERSITY

Overview

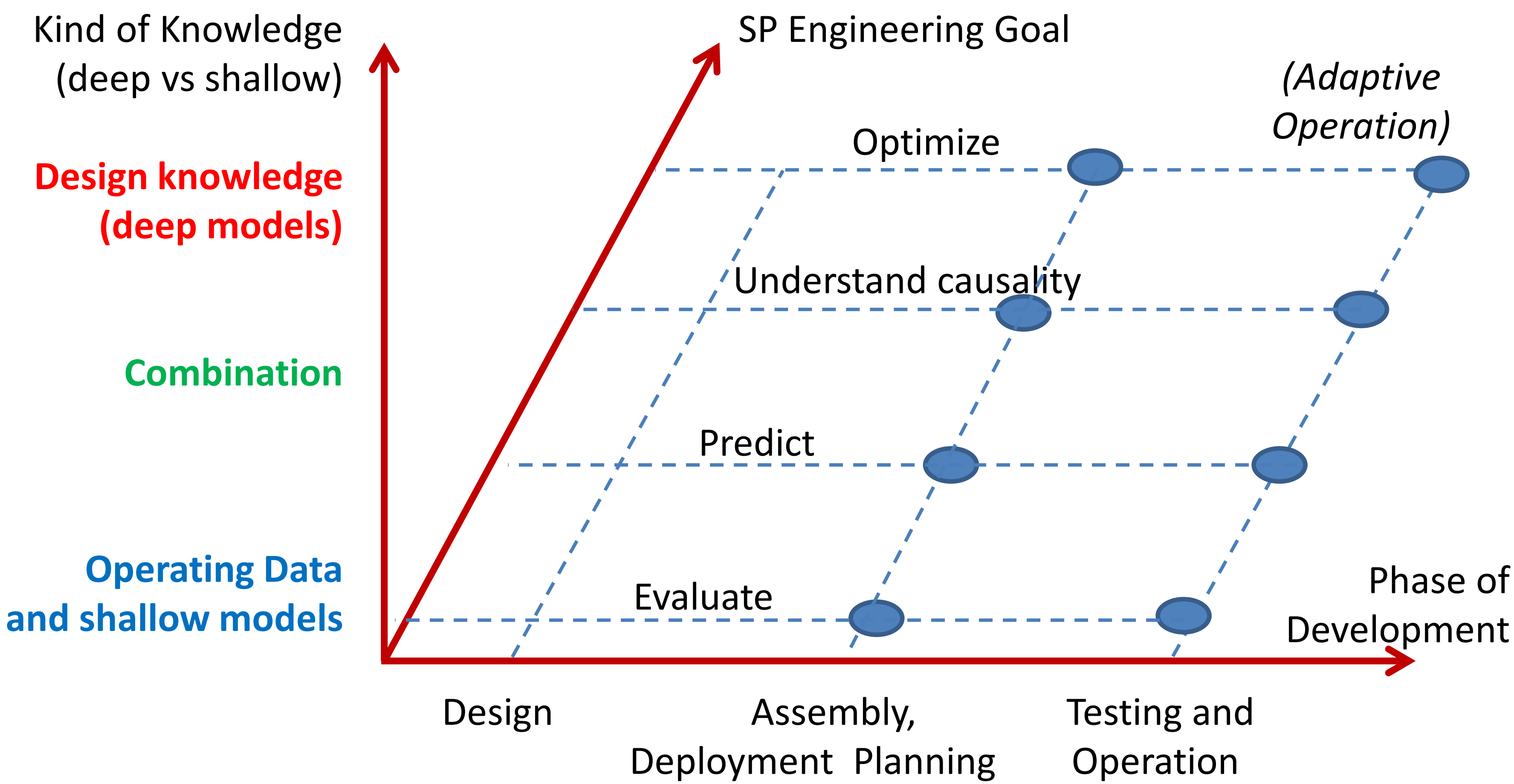
This will be a personal view….

- A perspective on the field
- Concerns
- Some challenges for current research
  - *Let's discuss as we go, particularly to identify additional points for later discussion.*

Three Dimensions of SP Engineering

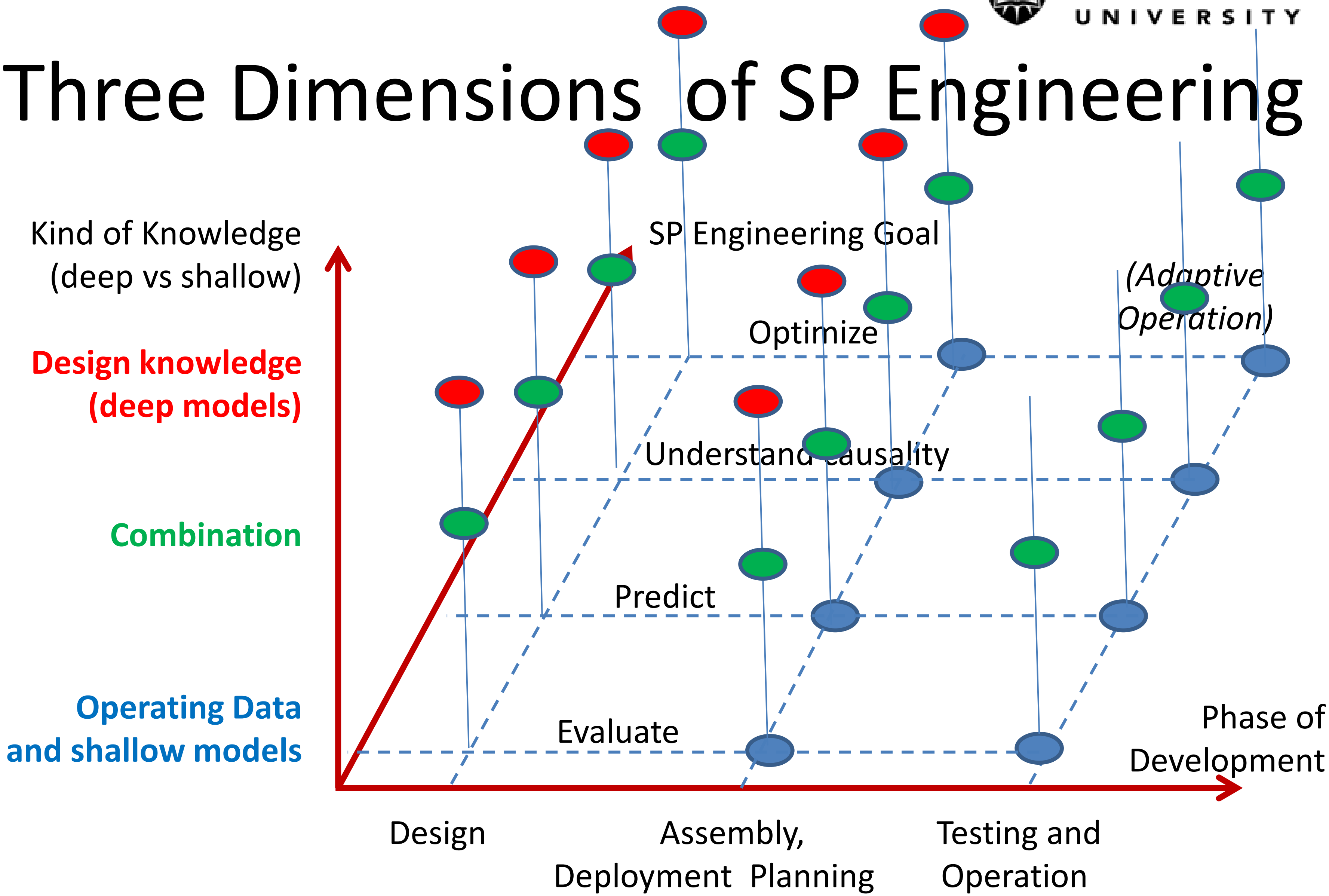


Three Dimensions of SP Engineering



Engineering with Operating Data (e.g. adaptation and "BigData" anaylsis)

Three Dimensions of SP Engineering



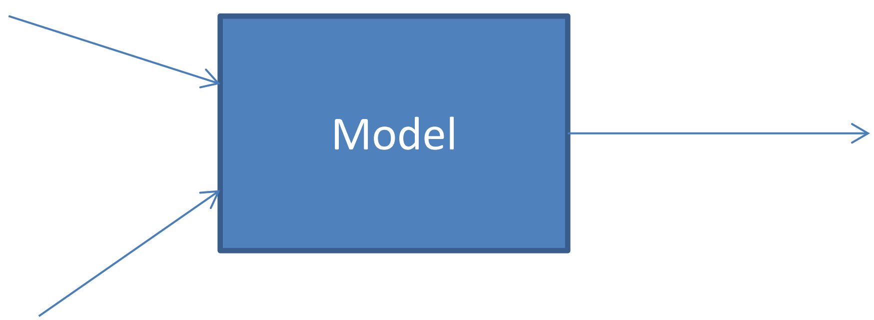
Including the Use of Deeper knowledge

The Sources of Knowledge…

- Data from both system logging and custom instrumentation
  - Including end-to-end tracing to follow a response
  - "Proactive" measurement is turned on all the time… operators now seem ready to do this
- Empirical modeling and Machine learning models (shallow modeling)
- Models based on the software design (deep models):
  - Automated extraction from design, such as a UML model
  - Structure extraction from deployment records
- Combined: learning models based on deep structure
  - Deep structure extraction (e.g. LQN from data traces)
  - Regression fitting of deep models such as queueing, layered queueing, SPNs or SPAs

A Spectrum of Models from Shallow to Deep

- I will use the term **Shallow models** for black-box models
  - inputs are load and configuration parameters and choices
  - outputs are performance measures



Shallow models have a **predefined structure** such as a polynomial or piece-wise linear function … e.g. a polynomial fitted by regression.

Spectrum… (2)

- **Deep models** have a mathematical form that represents the system structure, resources and behaviour (more or less exactly)
  - based on knowledge of the software and environment
    - Queueing and layered queueing models, Markov models
    - Stochastic Petri Net and Stochastic Process Algebra models
    - Simulations
- **Deep models are more robust and offer insight into causality**.
- **Fitted deep models** combine prior knowledge with parameters fitted to system observations
  - E.g. my 2008 paper on general non-linear regression methods for networks, Casale et al 2015 paper on fitting demands)

Spectrum (3)… Comments

- To progress, we should avoid religious adherence to one kind of model
- Ideally we want to exploit every crumb of knowledge
  - structure and measurements.
  - Fragments of knowledge
    - we may know parts of the structure but not all

Examples of Goals for "System Understanding"

- Detecting bottlenecks
- Detecting "long paths", execution paths that dominate the response time, including important latencies
- Scalability analysis
  - Finding elements that limit scalability
- Detecting performance antipatterns

"System Optimization" includes…

- Optimization of a software design
  - By trial and error
  - By diagnostic rules

- Optimization of an operational configuration
  - Planning an optimal configuration, using a model
  - Direct optimization by controlling parameters of the configuration
  - On-line use of remembered good configurations
  - On-line model-based optimization

Some of our Concerns

- Development and performance
- Design optimization
- Adaptation of software or deployment
- Combining deep models with measurement
- Application performance management

Carleton UNIVERSITY

Development and Performance

- Shift to assemblies of components and to web services means much of the performance is outside the designer's control
  - An opportunity to calibrate predictive submodels for rapid model assembly…. Not so easy
- Adaptive operation covers up for design deficiencies
  - Design effort goes into scalable designs, such as event-driven systems
- DevOps is the new route to performance-aware design
  - Casale presentation to 2015 WOSP-C
  - Industry (e.g. IBM) is pushing performance-aware DevOps

Design Optimization

Manual, testing-based

- Need for assistance in diagnosis (current work)

Manual techniques combined with models for evaluation

- Smith's design principles, such as early binding
- Performance antipatterns, such as the one-lane bridge

(Semi-) Automated techniques based on models

- Rule-based diagnostics
- Antipattern diagnosis

Operational Tuning and Adaptation

This is the central focus of companies creating systems…Application Performance Management is the mantra

- An active and developing market for tools
- Shift from enterprise systems to web-based
- “proactive” means continuous monitoring in depth
- On-line analysis implies use of shallow models
  - E.g. Zhu keynote to ICPE 2014 listed clustering, and learning models including regression
- Lots of opportunities to tie in design and prediction

Combining Deep Models with Measurements

In "The Future of Software Performance Engineering" (2007) I and co-authors emphasized **the need to combine modeling with data**, particularly to build models into data analysis

Since then we see integration in:

- the treatment of testing in Andre Bondi's book
- recent papers on "big data" methods for performance e.g., use of machine-learning models.
- the "Filling the Gap" tooling described by Casale et al at the last workshop.

- There is a deep problem here in obtaining generality.

- A PLUS: proactive monitoring is making better data available

Application Performance Management

- This is where system managers hope to deal with performance.
  - Zhu keynote at ICPE in Dublin: use of "big data" methods for APM
  - Machine learning of models was emphasized
- Recent survey of the APM tool market (Clabby) notes
  - A shift from managing enterprise systems, to web-based systems
  - A shift from technical metrics like response time to user experience (broader concerns)
  - Use of proactive monitoring, meaning on all the time, and data analytics

For Discussion:
Some Challenges and Opportunities

- Design and Performance
- Combining models and data
- Augmenting measurement
- Exploiting the full spectrum of model abstractions
- Performance antipatterns
- Dynamic architectures
- More??

Challenges in Design and Performance

- Today the **software wizards** have the upper hand over the **software engineer**s
  - Fast, intuitive, leave the problems (like performance) for others
  - In these projects, it is always “fix-it-later”.
    - So, effective SP research must be on improving existing systems
  - SP research has been focussed on the software engineers,
    - e.g. MDD for performance

- **??Can SP research also help the wizards?**
  - High-performance architectural templates?
  - Antipattern detection in source code?
  - Aggressive performance-related analysis of unit tests?
  - Model extraction from test, and model-based diagnosis?

Combining Models and Data

- We have the data..☺
- The need for abstractions to understand it is recognized

- So… how to go about it?
  - We should exploit knowledge of deep structure
  - A challenge: partial knowledge
  - One idea: base a structure on prior knowledge, fit parameters by nonlinear regression (essentially hill-climbing) methods
  - Maybe also: Search for resource effects in the data

Carleton UNIVERSITY

Models and Data (2)

- A strategy: Partition the fitting process for efficiency
  - Many configuration parameters only affect the CPU demand or some other known parameter in a prior structure
  - Many only affect a particular known component… fit submodels for greater accuracy

Models (3)... Making them Better

- Models have not achieved “top-of-mind” in development **because models are not widely enough trusted**.
  - I feel this is because of weaknesses in modeling.
- **The real challenge is to earn trust**.
- One possible answer to this challenge is to make model-making more transparent

- **Observation**: models fitted to data are trusted more than models created by analyst knowledge or by transformation of a design

Gaps in the Power of Measurement

- Widely available instrumentation is only strong for utilization and event-counting, and delays at a single point
  - Delay across a distributed system remains vexed
  - Recent survey on end-to-end tracing describes the state of the art
- Measurement systems have long needed standardized semantics
  - The Descartes metamodel addresses this concern

Exploiting the Full Spectrum of Model Abstractions

- Could we develop a capability to move incrementally from shallow to deep models?
  - Maybe by combining shallow and deep structure?

- Find a way to use fragments of structural knowledge

More Kinds of Performance Antipatterns

- As defined by Smith and Willams, these are properties of the design
  - their papers,
  - book and papers by Mirandola and Cortellessa,
  - ICSE 2014: Chen et al "Detecting Performance Anti-patterns for Applications Developed using Object-Relational Mapping".
- Measured data also displays the problems
  - A saturated resource, identified from its utilization
  - A function that takes too long, identified from profiling

So, could we detect the antipatterns from measurement?

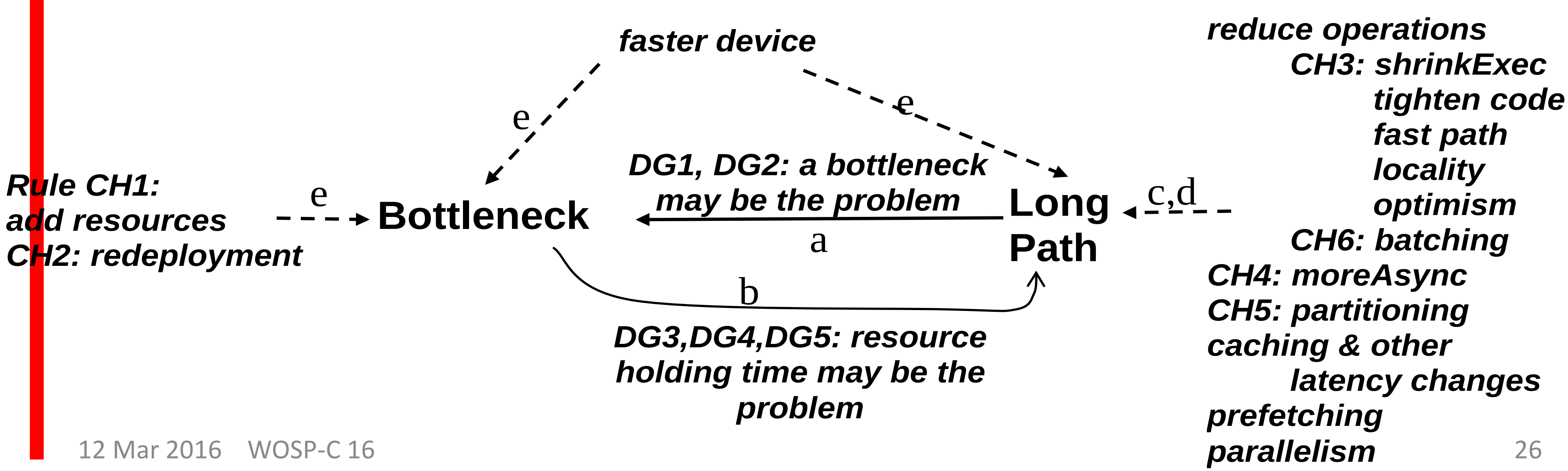
- e.g bad interprocess interactions

Carleton UNIVERSITY

Performance Antipatterns (2)

- antipatterns were classified into two groups by Jing Xu (WOSP07)
  - Bottleneck-creating antipatterns such as the one-lane bridge
  - Long sequential path antipatterns

and it was shown how their causality may be nested:



Challenge: Systems with Dynamic Architectures

Some big examples…

- Web service systems may be assembled dynamically and used for a little while, then reassembled
  - can be assembled on the fly for a single request
- Ad hoc networks of sensors and of mobile terminals
- Internet of Things

More detailed examples:

- Degree of parallelism may depend on the data being processed
- Reliability and performance-seeking adaptations modify architecture
  - Series of snapshots with static architecture

Dynamic Architectures (2)

- Some examples show variation within a framework
  - So… the framework is the static part of the architecture
  - Parameterize the dynamic changes can be parameterized
    - Number of parallel paths, choice of alternative servers
- We have so far worked by decomposition
  - Combine solutions of a set of possible “static” cases
  - But this loses temporal detail and the effect of transients

More…

- Simulation challenges
- Embedded systems and Sensor networks
- Better methods for performance requirements
- Intelligent design of performance stress tests

- And more…

References

- G. Casale, “Performance Aware DevOps”, presentation to WOSP-C 2015.
- M. Woodside, "The Relationship of Performance Models to Data", Proc SPEC Int Workshop on Performance Evaluation (SIPEW), Darmstadt, LNCS Vol. 5119, pp 9 - 28, June 2008 (*Regression fitting of any performance model, to performance data*)
- T. Chen, W. Shang, Z.M. Jiang, A.E. Hassan, M. Nasser, P. Flora, “Detecting Performance Anti-patterns for Applications Developed using Object-Relational Mapping”, ICSE 2014, Hyderabad, India. (*Develops special problem-dependent antipatterns)*
- A.B. Bondi, “Foundations of Software and System Performance Engineering”, Addison-Wesley, 2014. (*Bases test interpretation partly on properties of flows, utilizations and contention*).
- M. Woodside, G. Franks, D.C. Petriu, "The Future of Software Performance Engineering", Proc Future of Software Engineering 2007, part of ICSE 2007, May 2007, pp 171-187
- W. Wang, J.F. Pérez, G Casale, “Filling the Gap: a Tool to Automate Parameter Estimation for Software Performance Models”, Proc QUDOS 2015, 1st Int Workshop on Quality-Aware DevOps, pp 31-32 .
- R.R. Sambasivan, R. Fonseca, I. Shafer, G.R. Ganger So, you want to trace your distributed system? Report CMU-PDL-14-102. April 2014. (*Survey and critique, including end-to-end tools*)
- Jing Xu, "Rule-based Automatic Software Performance Diagnosis and Improvement", Proc. ACM WOSP 08, Princeton, June 2008 (*metric-based antipatterns, rules for model evolution*)
- Jane Clabby, “How the APM market has evolved to keep pace with changing application environments”, Computerworld, Jun 10, 2015. (*Commercial issues in APM*)