

# Runtime Assurance for Big Data Systems

John Klein

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Copyright 2015 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

DM-0002842

# System Measurement for Assurance at Runtime

Big Data systems have a very dynamic runtime context – assurance “by design” is still necessary but not sufficient

- New and changing data sources – sensors, humans, systems
- Evolving user workloads driven by new missions and new data
- Shared infrastructure – variable quality of service

Need to monitor the big data system in its runtime environment

- Collect measurements/metrics
- Assess health and trigger action to assure capability delivery



# System Measurement for Assurance at Runtime

Big Data systems have a very dynamic runtime context – assurance “by design” is still necessary but not sufficient

- New and changing data sources – sensors,
- Evolving user workloads driven by new mis
- Shared infrastructure – variable quality of serv

Focus of this year's project

Need to monitor the big data system in its runtime environment

- Collect measurements/metrics
- Assess health and trigger action to assure capability delivery

Future Work –  
Predict, Diagnose

# Technical Challenges – Measurement Collection in Big Data Systems

## System scale

- 1000s of nodes
- Millions of measurement time series streams
- Efficiency is critical

Processing and storage framework resiliency and redundancy makes individual node status less meaningful

- Need aggregate application-level measurements composed from component data
- End-to-end system performance is the ultimate health measure



# Solution Approach

## Scale → Automation

- Monitor generation
- Monitor insertion
- Measurement collection and aggregation
- (Future) Generate visualizations

## Architecture Styles → Abstractions and constraints to enable efficient automation (metamodels)

- Styles/Patterns capture common architecture approaches
  - Restrict types of components and topologies
  - Establish semantics for a class of functionality
  - Define what to measure, where to measure, how to aggregate



# Solution Approach

Scale → Automation

- Monitor generation
- Monitor insertion
- Measurement collection and aggregation
- (Future) Generate visualizations

New Contribution

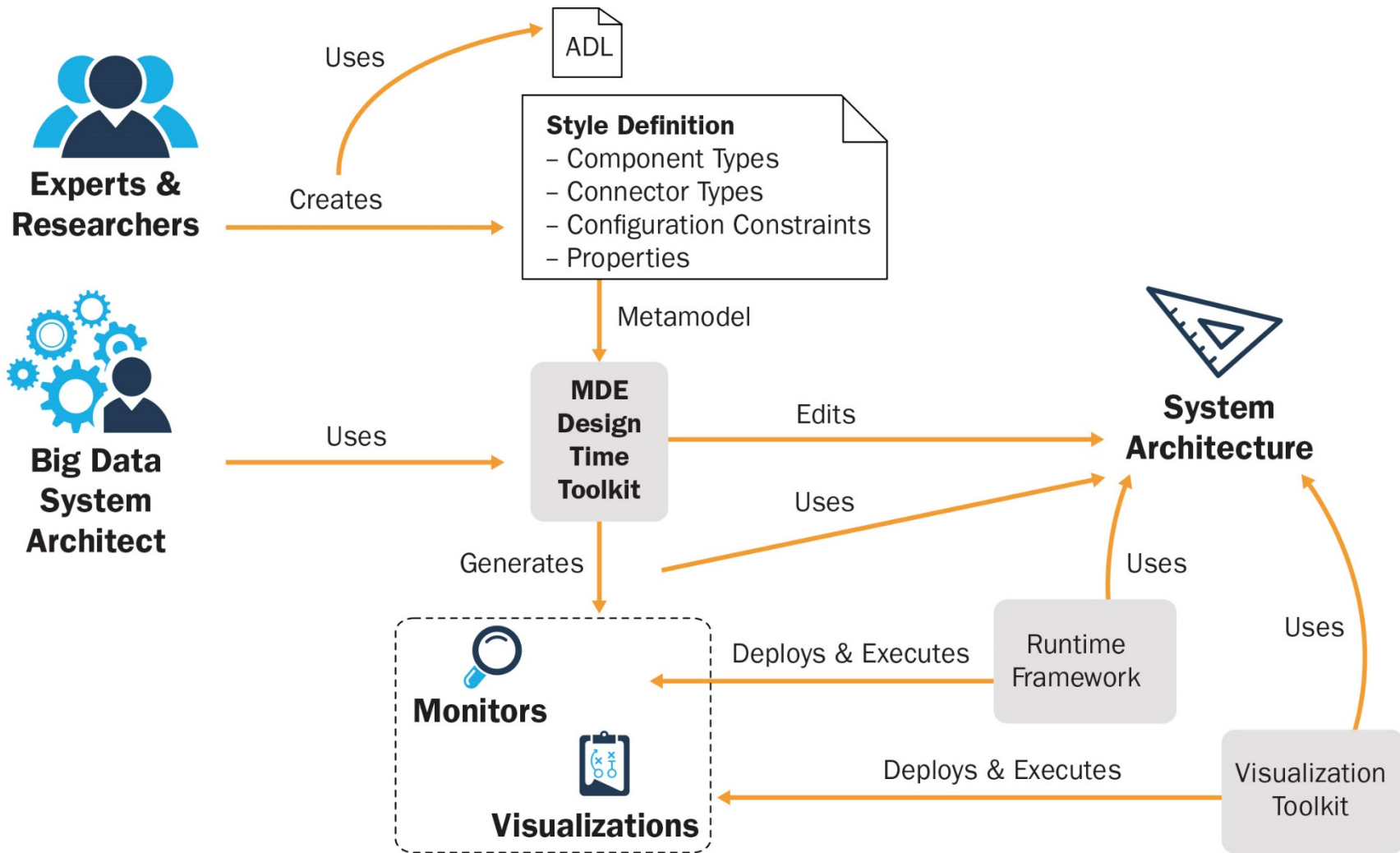
Architecture Styles → Abstractions and constraints to enable efficient automation (metamodels)

- Styles/Patterns capture common architecture approaches
  - Restrict types of components and topologies
  - Establish semantics for a class of functionality
- Define what to measure, where to measure, how to aggregate



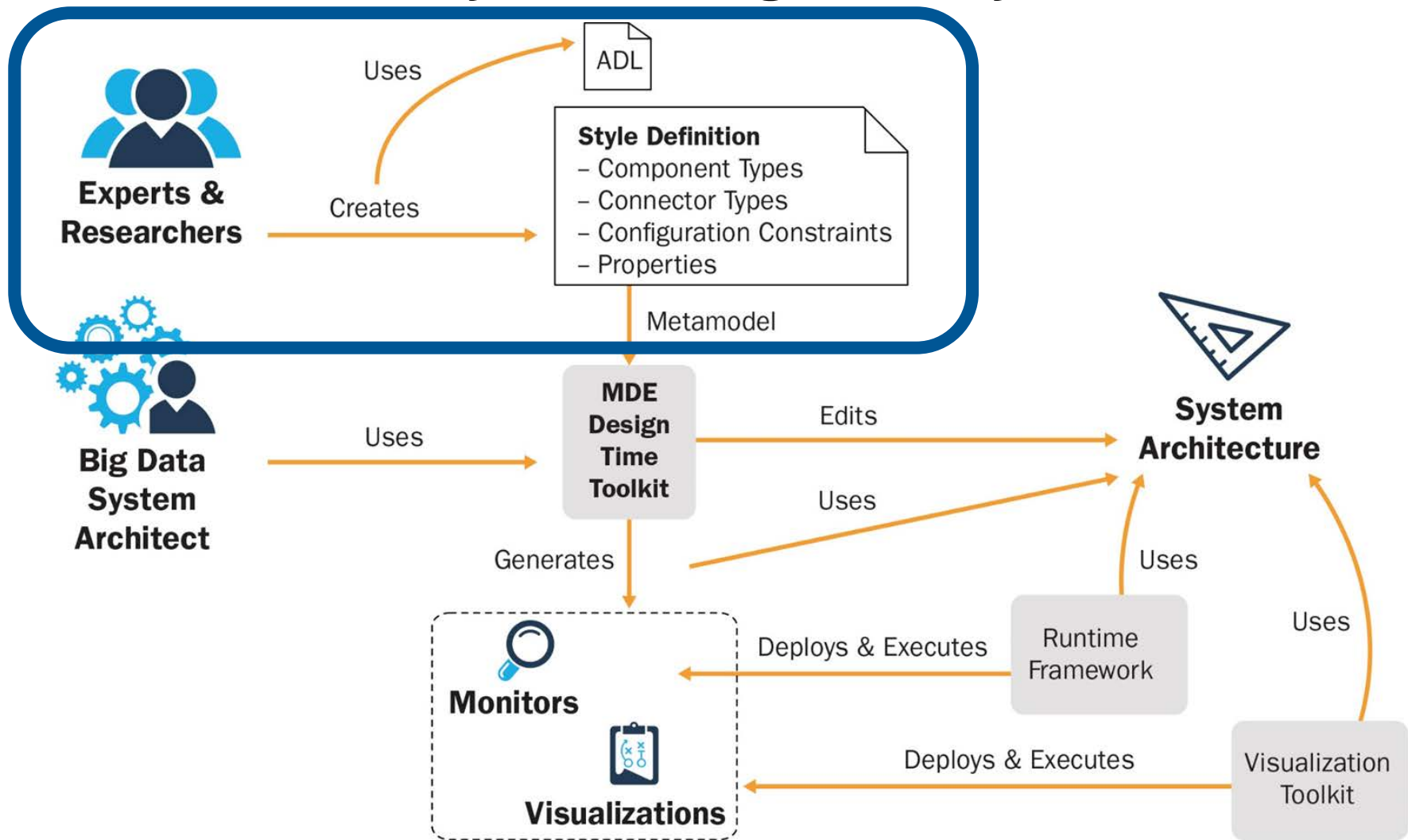
# Scale → Automation

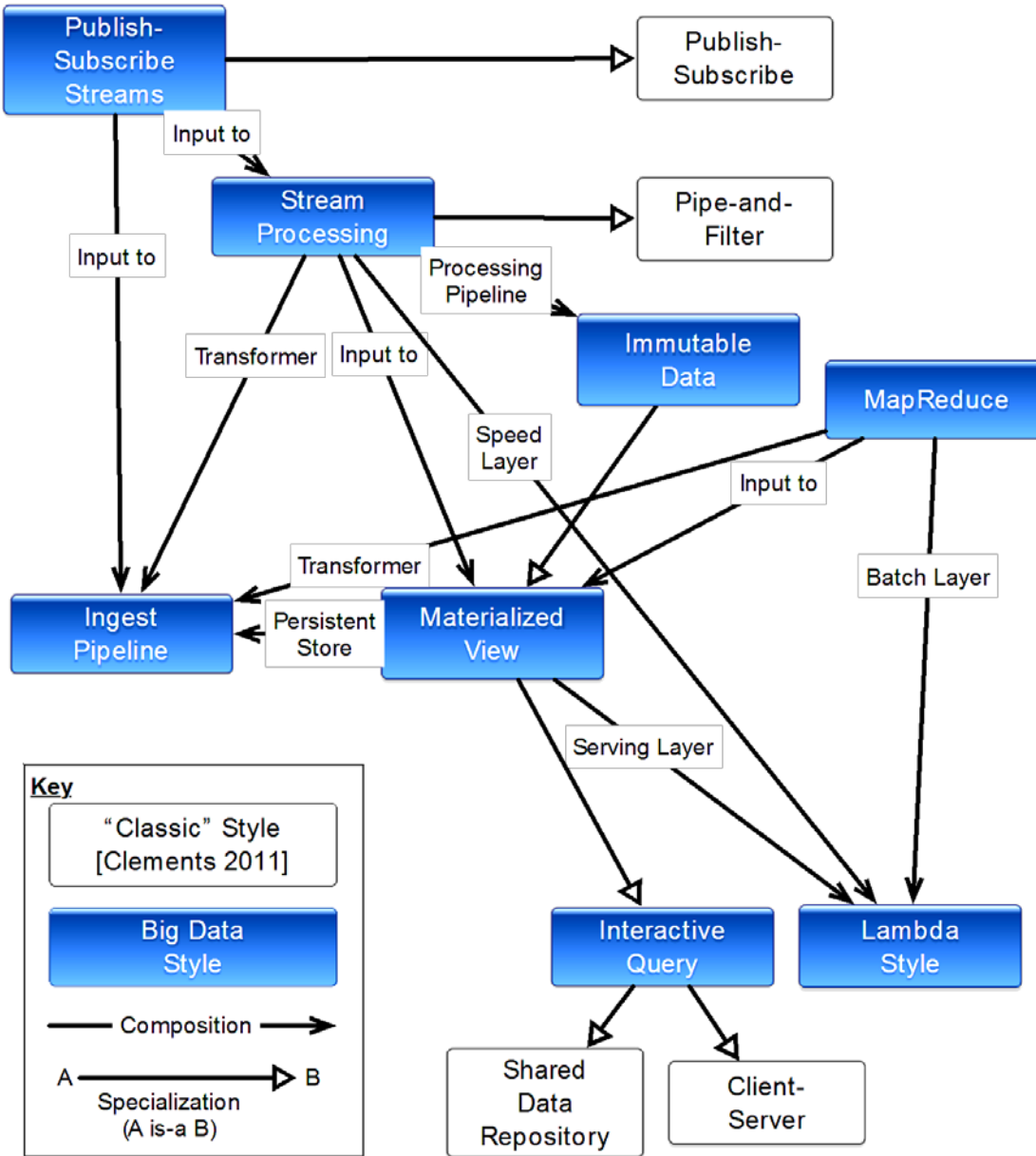
## Leverage Architecture Styles to Automate





# Results – Architecture Styles for Big Data Systems





# Architecture Styles for Data-Intensive Systems

# Example – Ingest Pipeline Style

## Component Types

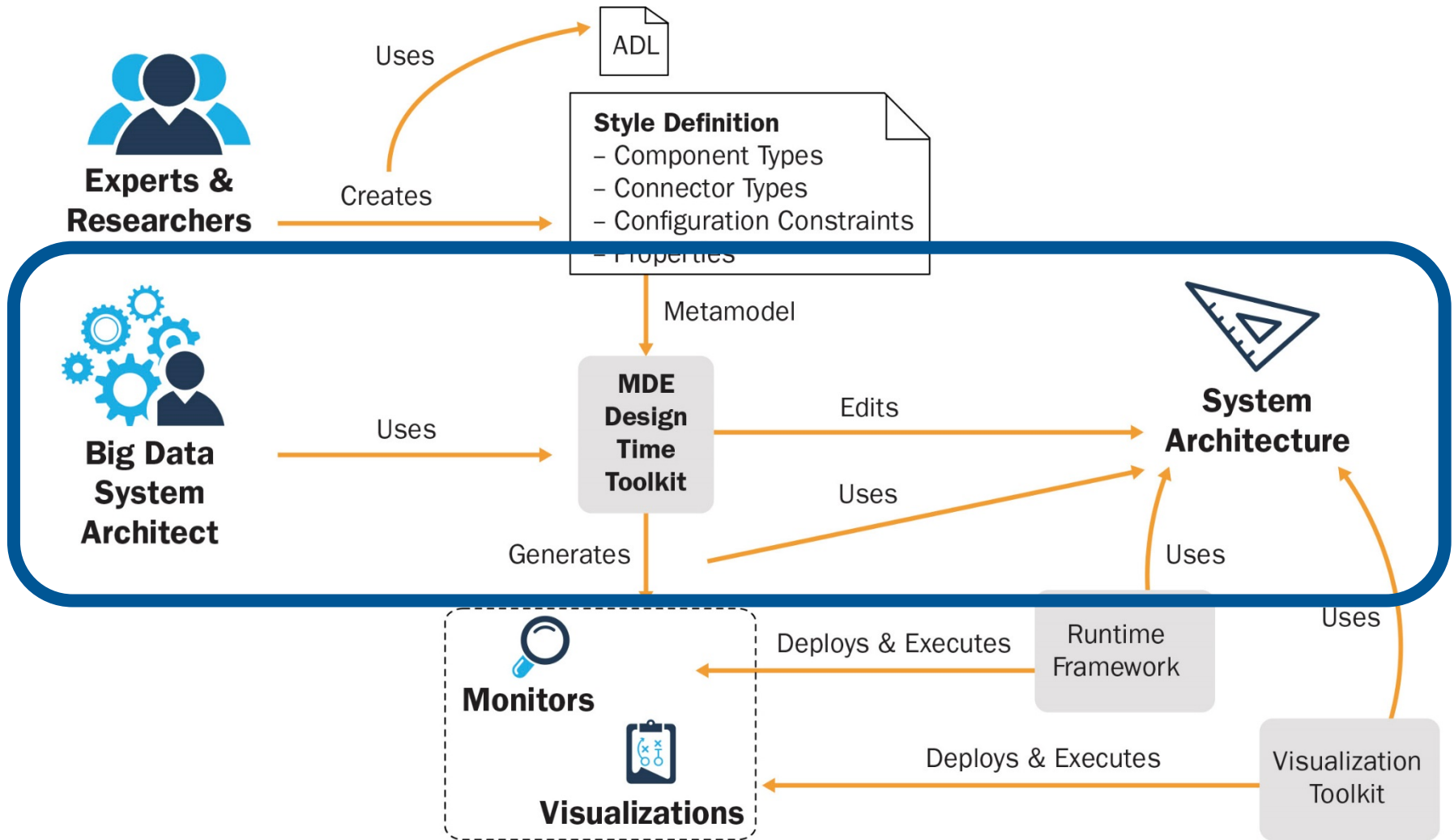
- Source Adapter, Transformer, Writer, Persistent Store

## Measurement Metamodel

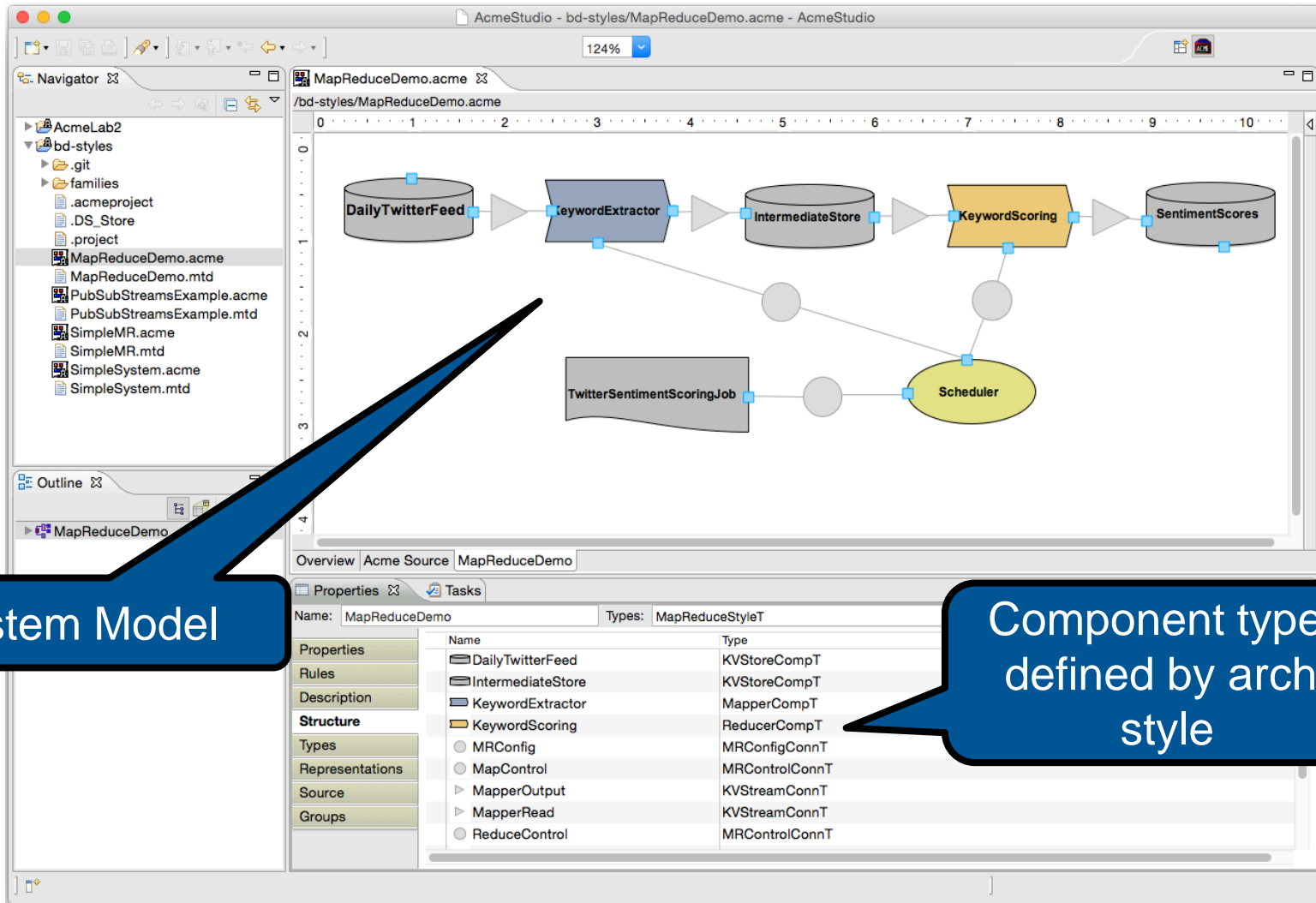
- A healthy instance of the *ingest pipeline style* processes data at a rate that keeps ahead of incoming data. The measurements to provide visibility into this include:
  - # of input messages/records and rate for each input source
  - # of data store writes per namespace and write rate
  - Application-specific counts of values of particular input and output data types (“peg counts” or histograms) to assess the distribution of input and output data sets.



# Results – Using Styles to Model Big Data Systems



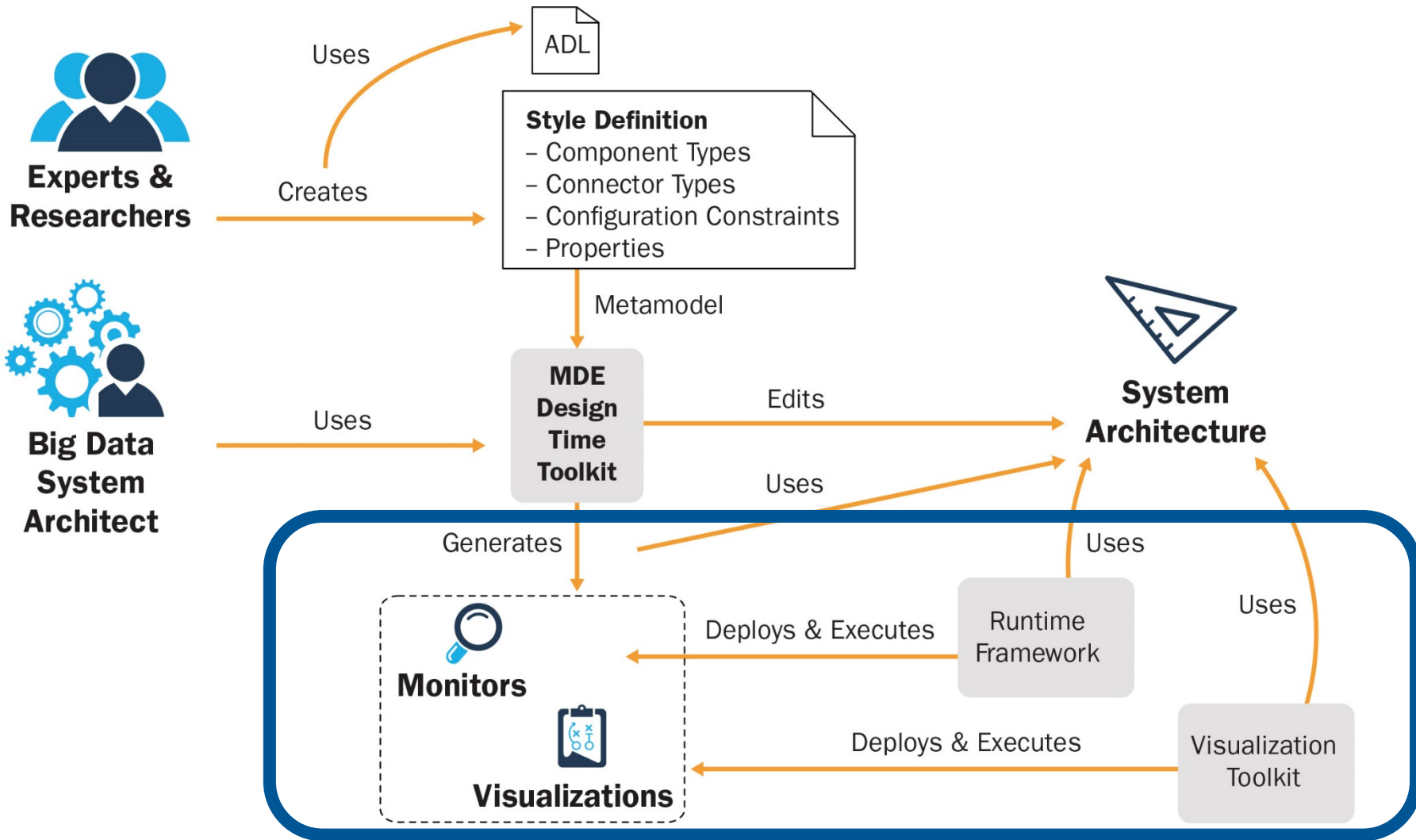
# Style-Based System Modeling in Acme Architecture Description Language



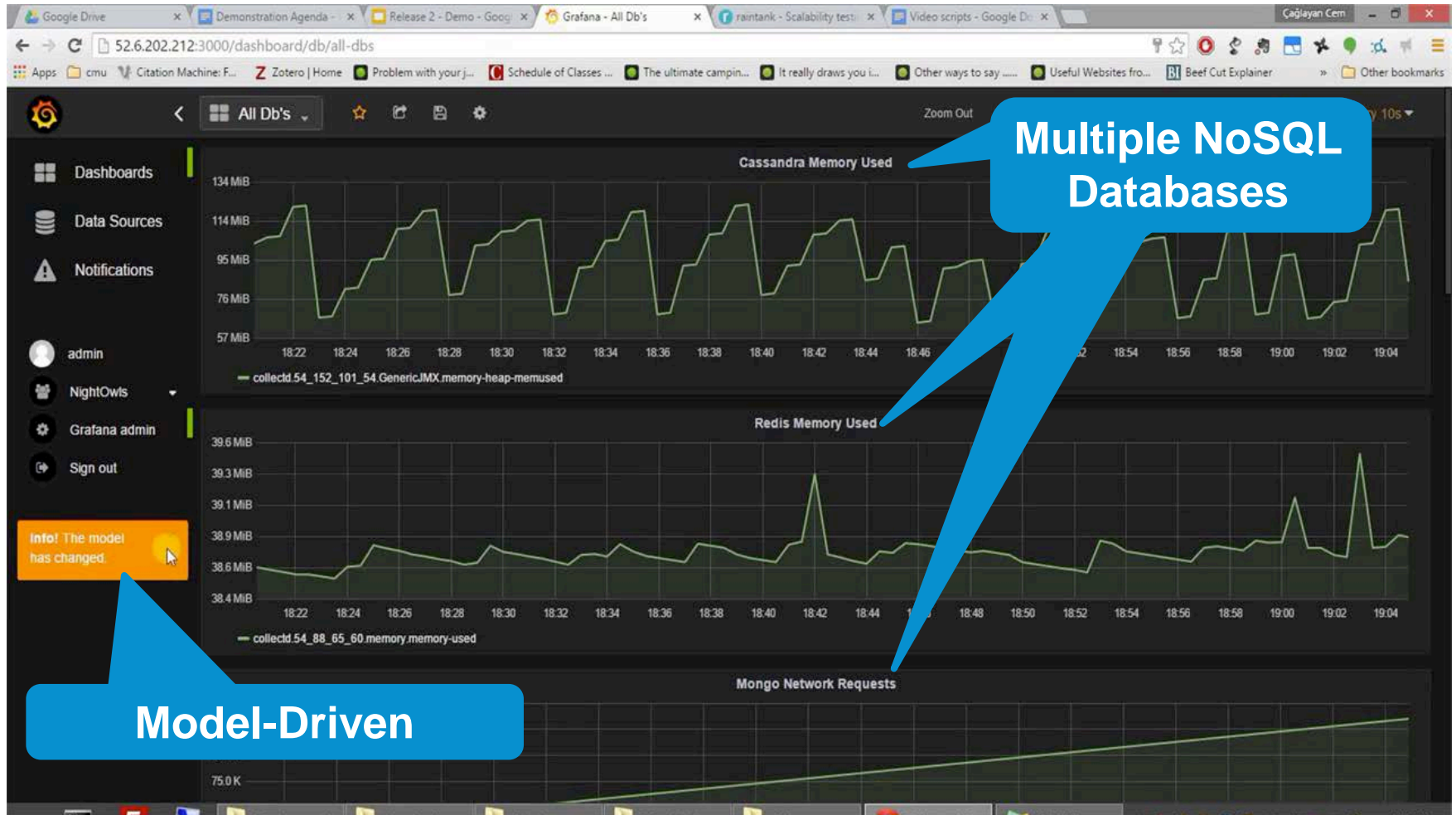
System Model

Component types defined by arch. style

# Results – Automated Monitor Generation and Measurement Collection



# Model-Generated Monitors and Collection for NoSQL Persistence (CMU MSE Project)



# Results and Future Work

Demonstrated feasibility of automating measurement collection and aggregation

- Steps are manually integrated
- Limited monitor generation

Future Work:

- Transparent monitor insertion into existing systems that use open source processing frameworks (e.g., NTC using Apache Storm)
- Extension of style catalog for machine learning systems
- Automated generation of advanced visualizations
- Anomaly detection analytics (e.g., analytics like Netflix “starts per second” analytics)





# Contact Information

## John Klein

Senior Member of Technical Staff  
Software Solutions Division  
Telephone: +1 617-283-2170  
Email: [jklein@sei.cmu.edu](mailto:jklein@sei.cmu.edu)

## Web

[www.sei.cmu.edu](http://www.sei.cmu.edu)  
[www.sei.cmu.edu/contact.cfm](http://www.sei.cmu.edu/contact.cfm)

## U.S. Mail

Software Engineering Institute  
Customer Relations  
4500 Fifth Avenue  
Pittsburgh, PA 15213-2612  
USA

## Customer Relations

Email: [info@sei.cmu.edu](mailto:info@sei.cmu.edu)  
Telephone: +1 412-268-5800  
**SEI Phone:** +1 412-268-5800  
**SEI Fax:** +1 412-268-6257