Statement of Research

Aniruddha S. Gokhale (a.gokhale@vanderbilt.edu)
Associate Professor, Dept. of EECS
Vanderbilt University, Nashville, TN 37235

July 29, 2017

1 Executive Summary

At its core, my research blends Software Engineering principles with Systems Research. In this context currently I am focusing on developing novel algorithms and software-defined techniques to solve a myriad of distributed systems challenges in realizing resilient cyber physical systems. Specifically, the contours of my current research involve addressing challenges in mobile and edge cloud computing, which in turn includes (a) dynamic resource management based on dynamic system model learning, simulation and system actuation, (b) reconciling real-time stream processing – which is required for the distributed Big Data analytics, with data-centric publish/subscribe systems – which is required for information-centric networking, and (c) algorithms and mechanisms for software-defined, resilient infrastructures – which are required to manage and safeguard system resources and applications. An outreach aspect of my current work focuses on using cloud computing technologies for STEM education using real-world problems from my current research as challenge problems for STEM education and collaborative learning. My ongoing and planned research, as presented in this document, has been informed by emerging trends and insights gained from work to date.

2 Emerging Trends and Research Vision

The proliferation of smart, mobile endpoints including sensors and smartphones that form the genesis of the Internet of Things (IoT) paradigm, and the ensuing massive amounts of Machine-to-Machine (M2M) communications, has created a myriad of challenges that manifest in the next-generation of cyber Physical Systems (CPS), such as Smart and Connected Cities, Smart Grids and Smart Manufacturing. Addressing these challenges forms my research agenda over the next five years.

To understand these challenges, consider how these CPS are increasingly supporting device-to-cloud as well as cloud-to-cloud communication patterns for scalable data analytics and fault tolerance requirements [41, 45]. Thus, the communication networks for these CPS must interconnect a variety of smart objects that are designed for different hardware platforms, support large-scale deployment of IoT devices, and handle a massive number of generated events. Interconnecting heterogeneous smart objects over the Internet is, however, a very difficult goal to achieve because these CPS applications operate in diverse environments – often remote and hostile – and involve different functional and quality of service (QoS) requirements. Provisioning and sharing the communication links across the CPS for the diverse needs of the different traffic flows remains an unresolved problem.

At the same time, due both to the resource-constrained nature of the system (e.g., available communication bandwidth and battery power) and low latency requirements of many of the CPS applications (e.g., driver notification in a smart transportation system), the data analytics (e.g., analyzing traffic congestion data in a segment of a road network) cannot always be performed in a centralized cloud, which may be many network hops away from the source of the data and can cause substantial delay in the results propagating back. Consequently, the data analytics must be performed in a distributed manner and often closer to the source. The emergence of cloudlets [123], which are akin to micro data centers and are closer to the source, as well as the increasingly powerful edge devices make edge/cloudlet computing appealing to address these challenges.

Despite this promise, the outcomes of the individualized and localized analytics may not be sufficient to obtain an accurate operating picture of the overall system, which is needed for the effective management and resilience of the system through a range of actions, such as dynamic management and scheduling of resources, and reconfiguration and redeployment of applications. Consequently, information-centric networking solutions that process the deluge of information in real-time and in a distributed manner opportunistically at distributed resources ranging from the edge to the cloud [134], and which can disseminate the outcomes to interested parties over wide area networks [6] are needed.

My research agenda is addressing these intertwined set of challenges holistically through the following primary set of synergistic research efforts:[1]

- **Distributed, real-time stream processing blended with wide-area publish/subscribe (pub/sub):** New research directions that will realize WAN-scale pub/sub for CPS that provide control over the communication links, and which enable both edge- and cloud-based analytics is needed. In this context, we focus on data-centric pub/sub since it provides the desired semantics for event-based interactions ensuring scalable and decoupled data sharing mechanisms between communicating peers within CPS [6].

  Coupled with this information dissemination focus, we are investigating mechanisms to execute real-time stream processing tasks for model learning opportunistically across the spectrum of resources ranging from the edge to the cloud. In doing so, our aim is to reconcile the real-time needs of stream processing with that of real-time dissemination. Together, this approach will help to develop a global system model that can be used for system management and resilience [82].

- **Dynamic resource management:** To enable any of the earlier research foci, effective dynamic resource management across the spectrum of resources is required. Such a system-level resource management requires effective models of the system, which must be learned online and subsequently used to control and schedule the resources [25, 29, 134]. This dimension of the problem is different from the application-level analytics. To that end, the software-defined approaches that separate the control plane from the data plane are investigated so that system-level activities including analytics and dissemination are performed at the control plane such that any changes to the algorithms and policies can impact only the control plane while not impacting the data plane where the applications reside [65, 106, 62, 63, 105].

- **Performance optimizations and system resilience:** A consistent theme of my research right from my doctoral research has been about performance optimization and resilience. Proper and principled implementation of the devised algorithms is necessary to realize elegant and extensible implementations that make optimal use of resources; this issue is all the more important for resource-constrained parts of the CPS we consider in our research [27, 87, 23]. Similarly, system resilience is another requirement for the CPS where human-driven control may not be feasible at all times. Thus, my existing collaborative research efforts are exploring the use of goal-oriented, autonomous system (re)configuration and (re)deployment to make them resilient [112, 113, 111].

[1]Individual problem areas of research and resulting contributions are not discussed in this document; instead these can be obtained from my publications listed in my CV.

These research ideas are informing my agenda for education and outreach. For example, conducting these research investigations in the context of current projects, e.g., NSF US Ignite's software-defined networking project, Siemens' Edge computing project, AFOSR DDDAS's dynamic model learning project, and Vanderbilt TIPS' Smart cities project, are providing me with real-world scenarios and challenge problems that I am using to further improve the web-based STEM learning environment called C^3STEM that we have built as part of an NSF project and have been using to conduct user studies with high school students from the Nashville metropolitan public schools [36, 133, 37, 28]. Moreover, the harder challenge problems also serve as interesting project topics for my Cloud Computing (CS 4287/5287) and Distributed Systems (CS 6381) courses. These also will form the basis of a Distributed Systems Specialization MOOC that I plan to co-offer on the Coursera platform in Fall 2018.

The rest of this document presents my research contributions in reverse chronological order. This order was deemed necessary since the most recent research provides the most compelling arguments for my current and planned research directions. Section 3 presents insights gained from research conducted post promotion and tenure; Section 4 presents a summary of my research during my years as a tenure track faculty; Section 5 presents a summary of my industrial research; Section 6 presents a summary of my doctoral research; and finally Section 7 discusses my research plan including current and planned collaborations within and outside of Vanderbilt University.

3 Contributions and Insights from Research Conducted as Associate Professor (Sept 2010 – present)

The research I have been involved in after being tenured and promoted to Associate Professor has been instrumental in shaping up the ideas for my ongoing and planned research agenda. This recently concluded research can be classified along the following general themes and dimensions, all applied to supporting various properties of cyber physical systems: (a) Resource Management and Autoscaling for the Cloud infrastructure, (b) Software-defined Networking solutions to control networking resources, (c) Parallel dataflow models for in-network data analytics and dissemination, (d) Middleware specialization, and (c) Deployment and configuration heuristics. A major portion of my contributions during this time were funded by my NSF CAREER award, which I had secured a year prior to tenure, by AFOSR DDDAS award, and recently also by the NSF US Ignite award.

3.1 Cloud Infrastructure Resource Management to Support CPS Applications

Contemporary and next-generation of CPS need to process significant volumes of data, which are generated by a heterogeneous set of sources, e.g., mobile devices, social media, and a number of the sensors. In the next five years, it is expected that mobile traffic will have grown thirteen times more than the existing mobile traffic and there will be three times more connected devices than the number of people on the Earth [74]. Similarly, scientific experiments such as CERN also generate enormous amounts of data estimated to be about twenty-five petabytes in a year [99]. With the emergence of the IoT paradigm, billions of data points are generated and as a result, the volume of this data is getting even larger.

All of this generated data must be processed to extract useful features out of it. This growing, massive amounts of data require more storage and compute resources, which is ultimately provided by the data centers throughout the world and the cloud computing infrastructure. As more and more applications are created, the cloud computing in general and data center in particular have become critical for many projects, enterprises, and research communities. Hence, it will continue to play a crucial role in delivering a variety of services.

Despite the fact that there is a significant momentum towards moving to the cloud, a variety of issues still exist in utilizing the cloud to its fullest potential. For example, energy efficiency, capacity planning, performance management, disaster management, and security are a few major concerns faced by cloud service providers (CSPs) among others. The energy consumption of data centers worldwide has reached staggering proportions and this trend will further continue. Moreover, diesel power generators, due to power outages in data centers and power plants, emit millions of tons of carbon [145, 84]. Thus, CSPs must address energy efficiency issues for data centers. At the same time, meeting the timeliness and reliability of CPS applications executing in the cloud is paramount. To that end, my research has addressed the following challenges in the context of enabling the cloud platforms for CPS applications.

3.1.1 Challenge 1: Autonomous and Dynamic Scheduler Reconfiguration

At the virtualization layer of a data center, hypervisors have a scheduling mechanism to deal with sharing CPU resources among the virtual machines (VMs) and executing the workloads in the VMs. Relying on default values, manually tuning the scheduler's parameters by following known configuration patterns, using generally accepted rules, and adopting trial-and-error approach, are common practices among the system administrators of the cloud data center. However, these approaches are not effective and efficient, particularly when dealing with dynamically changing workloads on the host machines and varied CPU resource utilizations. Moreover, these non-scientific approaches do not consider the resource overbooking ratios for resource management. Furthermore, often these manual decisions are made offline, which invariably cannot consider the overall system dynamics leading to poor system performance. Therefore, an online, autonomous, and self-tuning system for scheduler configuration is desired.

To address this challenge, we have developed iTune [27], which is a middleware that optimizes the Xen hypervisor's scheduler configuration parameters autonomously through a three phase design workflow comprising: (1) Discoverer, which monitors and saves the resource usage history of the host machines and groups set of related host machine workload, (2) Optimizer, where optimum Xen scheduler configuration parameters for each workload cluster is explored by employing a simulated annealing machine learning algorithm, and (3) Observer, where iTune monitors the resource usage of host machines online, classifies them into one of the categories found in the Discoverer phase, and loads the optimum scheduler parameters determined in the Optimizer phase.

3.1.2 Challenge 2: Resource-Overbooking to Support Soft Real-time Applications

Under-utilization, wastage of resources, and inefficient energy consumption are among the traditional issues of crucial importance to data centers. CSPs often overbook their resources by utilizing the tools in the cloud management layer. Overbooking is an attractive strategy to CSPs because it helps to reduce energy consumption and increase resource utilization in the data center by packing more user jobs in a fewer number of resources while improving their profits. Overbooking becomes feasible because cloud users tend to overestimate their resource requirements, utilizing only a fraction of the allocated resources. Without overbooking, resources in a data center will otherwise remain under-utilized.

Resource overbooking ratios are generally determined sporadically by analyzing the historic resource usage of workloads or following the best practices. Unfortunately, governing cloud resources in this manner may be detrimental and catastrophic to soft real-time applications running in the cloud. To make systematic and online determination of overbooking ratios such that the quality of service needs of soft real-time systems can be met while still benefiting from overbooking, there is a need for more efficient, effective, and intelligent approaches to overbooking that will ensure good performance for soft real-time applications yet prevent under utilization and also save energy costs.

To address this challenge, we have developed iOverbook [25], which is an overbooking strategy that uses a machine learning approach to make systematic and online determination of overbooking ratios such that the quality of service needs of soft real-time systems can be met while still benefiting from overbooking. Specifically, iOverbook utilizes historic data of tasks and host machines in the cloud to extract their resource usage patterns and predict future resource usage along with the expected mean performance of host machines. To evaluate our approach, we have used a large usage trace made available by Google of one of its production data centers.

3.1.3 Challenge 3: Performance Interference Effects on Application Performance

Although resource overbooking in the cloud is standard practice, it can lead to performance interference and anomalies among the VMs hosted on the physical resources, causing performance unpredictability for soft real-time applications hosted in the VMs. Such unpredictability may be detrimental to the performance of CPS applications that are controlled by the models executing in the cloud infrastructure. Moreover, resource overbooking can propagate and trigger faults in other VMs, which is also not acceptable to CPS applications. To address these problems and because workloads of the VMs may change at run time, virtual machine migration between physical host machines and data centers is the generally accepted mechanism.

Choosing the right set of target physical host machines for VM migration decisions plays a critical role in determining the performance and interference effects post migration. Analyzing the performance anomalies that might occur and predicting performance interference and fault before a VM is deployed or migrated on the physical host machines is thus desired and vital for soft-real time applications.

To address these issues, we have developed iSensitive [29], which is a machine learning-based middleware providing an online placement solution where the system is trained using events and lifecycle of a publicly available trace of a large data center owned by Google. Our approach first classifies the VMs based on their historic mean CPU, memory usage, and network usage features. Subsequently, it learns the best patterns of collocating the classified VMs by employing machine learning techniques. These extracted patterns document the lowest performance interference level on the specified host machines making them amenable to hosting applications while still allowing resource overbooking.

3.1.4 Challenge 4: Power- and Performance-Aware Virtual Machine Placement

Virtual machines are migrated from one physical host machine to another one in the same data center or across the data centers located in different locations due to fault tolerance, balance workload, application performance management concerns, and eliminate hotspots. Apart from the performance interference aspects described above, power and performance trade-offs are also critical and challenging issues faced by CSPs while managing their data centers. On the one hand, CSPs strive to reduce power consumption of their data centers to not only decrease their energy costs but also to reduce adverse impact on the environment. On the other hand, CSPs must deliver performance expected by the applications hosted in their cloud data centers in accordance with predefined Service Level Objective (SLOs). Not doing so will lead to loss of customers and thereby major revenue losses for the CSPs.

Power management and performance assurance are conflicting objectives, particularly in the context of multi-tenant cloud systems where multiple VMs may be hosted on a single physical server. The problem becomes even harder when soft real-time applications are hosted in these VMs. Solutions to address the virtual machine placement decisions exist. Bin packing heuristics such as first-fit, best-fit, and next-fit are common practices used by cloud management platforms (e.g., OpenNebula, OpenStack, etc.) to deploy VMs in the cloud. However, these solutions do not consider application performance and energy efficiency. To address the aforementioned issues, a power and performance-aware virtual machine placement algorithm is desired.

To address this challenge, we have developed iPlace [26], which is a middleware providing an intelligent and tunable power- and performance-aware VM placement capability. The placement strategy is based on a two-level artificial neural network, which predicts (1) CPU usage at the first level, and (2) power consumption and performance of a host machine at the second level that uses the predicted CPU usage. The placement decision (i.e., aptly suited host machine for the VM being deployed) is determined by making the appropriate trade-offs between predicted power and performance values of a host machine.

3.1.5 Challenge 5: Supporting Stochastic Hybrid Models of DDDAS Applications

With the advent of the Internet of Things (IoT) paradigm [13], there is no dearth of collected data. When coupled with technology advances in mobile computing and edge devices, users are expecting newer and different kinds of services that will help them in their daily lives. For example, users may want to determine appropriate temperature settings for their homes such that their energy consumption and energy bills are kept low yet they have comfortable conditions in their homes. Other examples include estimating traffic congestion in a specific part of a city on a special events day.

Deploying these services in-house is unrealistic for the users since the models of these systems are quite complex to develop. Some models may be stochastic in nature, which require a large number of compute-intensive executions of the models to obtain outcomes that are within a desired statistical confidence interval. Other kinds of simulation models require running a large number of simulation instances with different parameters. Irrespective of the simulation model, individual users and even small businesses cannot be expected to acquire the needed resources in-house. Cloud computing then becomes an attractive option to host such services particularly because hosting high performance and real-time applications in the cloud is gaining traction [3, 91].

Running these simulations sequentially is not a viable option as user expectations in terms of response times have to be met. Hence there is a need for a simulation platform where a large number of independent simulation instances can be executed in parallel and the number of such simulations can vary elastically to satisfy specified confidence intervals for the results.

To that end we have architected a cloud-based solution comprising resource management algorithms and middleware called Simulation-as-a-Service (SIMaaS) [135]. SIMaaS can elastically and on-demand execute the multiple different simulation trajectories of the simulation models in parallel, and perform aggregation such as stochastic model checking (SMC) to obtain results within a desired confidence interval. SIMaaS leverages Linux container [88]-based infrastructure, which has low runtime overhead, higher level of resource sharing, and very low setup and tear down costs. It provides a resource management algorithm, that reduces the cost to the service provider and enhances the parallelization of the simulation jobs by fanning out more instances until the deadline is met while simultaneously auto-tuning itself based on the feedback. The SIMaaS middleware intelligently generates different configurations for experimentation, and intelligently schedules the simulations on the Linux container-based cloud to minimize cost while enforcing the deadlines.

3.2 Software-defined Networking Approaches to Control Communication Resources

Software-Defined Networking (SDN) has emerged as a new intelligent architecture for network programmability. It moves the control plane outside the switches to enable external centralized control of data through a logical software entity called controller. The controller offers northbound interfaces to network applications that provide higher level abstractions to program various network-level services and applications. also uses southbound interfaces to communicate with network devices. One complementary technology to SDN called Network Function Virtualization (NFV) has the potential to dramatically impact future networking by providing techniques to refactor the architecture of legacy networks by virtualizing as many network functions as possible. NFV advocates the virtualization of network functions as software modules running on standardized IT infrastructure (like commercial off-the-shelf servers), which can be assembled and/or chained to create services.

Beyond traditional data centers where SDN is most applicable, the emergence of the IoT paradigm has given rise to many challenges. From a communications perspective, a high-level architecture of IoT systems is typically composed of three domains: Device Domain, Network Domain and the Application Domain. In the device domain, the device provides direct connectivity to the network domain via access networks, which may include limited range PAN technologies such as Bluetooth, ZigBee, etc., or via a gateway that acts as a network proxy for the network domain. Such a gateway must be flexible enough to efficiently manage available resources, QoS, security, as well as multimedia data exchange. These gateway concepts are prevalent in home ADSL models and WiFi access points found in cyber cafes and wireless hotspots. Because IoT systems integrate heterogeneous smart objects, the design of the gateway is quite different because it must not require each IoT subnetwork to have its own gateway. Thus, a convergent architecture towards a unique solution that integrates traffic incoming from heterogeneous smart devices should be designed.

Furthermore, as smart objects are resource- and energy-constrained, the gateway should be aware of the context of each process being managed. It should also employ intelligent routing protocols and caching techniques to route the traffic across the less constrained paths. The network domain includes different access networks, which provide connectivity through diverse technologies, such as xSDL, Satellite, etc, to devices and/or gateways. It also provides connectivity to the core network that includes heterogeneous and multi-technology connectivity, such as 3GPP, TISPAN, and LTE-A. Finally, the application domain includes the IoT applications and server/cloud infrastructures. The latter have to share their content, possibly back them up to other devices, analytic programs, and/or people who need to monitor real-time response to events. They also include service capabilities, which provide functions shared between different applications through open, high-level abstractions and interfaces that hide the specificities of the underlying networks.

My recent NSF US Ignite award is supporting my research investigations in the area of software defined networking as applied to addressing the myriad of challenges in traditional data centers as well as in wireless IoT environments. This dimension of research is still evolving and the following contributions have been made since early 2015.

- Articulated several different open challenges in SDN for both wired and wireless networks [65].
- Proposed an approach for bootstrapping the controller functionality in a project called InitSDN [106].
- Proposed an approach to use data-centric publish/subscribe as the messaging layer for the SDN northbound communications for IoT applications [62].
- Proposal a design of low rate wireless personal access networks (LR-WPAN) IoT Systems using SDN [64].
- Proposed an approach to use data-centric publish/subscribe as the messaging layer for a scalable southbound layer in SDN [63].
- Proposed an approach for data dissemination in wireless mesh smart grids using SDN [**?**].
- Proposed an approach to realize adaptive SDN-based multicast for group communications in data centers (paper in submission [107]).
- Proposed an approach for SDN-enabled wireless fog network management (paper in submission [66]).
- Proposed an approach for enabling SDN approach to be used in wireless mesh networks using a three-stage routing protocol [105].

3.3 Parallel Dataflow Models for In-Network Analytics and Dissemination

Critical CPS, such as smart-grids, intelligent transportation systems, advanced manufacturing, health-care tele-monitoring, etc, share several key cross-cutting aspects. First, they are often large-scale, distributed systems comprising several, potentially mobile, publishers of information that produce large volumes of asynchronous events. Second, the resulting unbounded asynchronous streams of data must be combined with one-another and with historical data and analyzed in a responsive manner. While doing so, the distributed set of resources and inherent parallelism in the system must be effectively utilized. Third, the analyzed information must be transmitted downstream to a heterogeneous set of subscribers. In essence, these critical systems can be understood as a *distributed asynchronous dataflow*. The key challenge lies in developing a dataflow-oriented programming model and a middleware technology that can address both *distribution* and *asynchronous processing* requirements adequately.

The distribution aspects of dataflow-oriented systems can be handled sufficiently by data-centric publish/subscribe (pub/sub) technologies [40], such as Object Management Group (OMG)'s Data Distribution Service (DDS) [100]. DDS is an event-driven publish-subscribe middleware that promotes asynchrony and loose-coupling between data publishers and subscribers which are decoupled with respect to (1) *time* (*i.e.*, they need not be present at the same time), (2) *space* (*i.e.*,they may be located anywhere), (3) *flow* (*i.e.*, data publishers must offer equivalent or better quality-of-service (QoS) than required by data subscribers), (4) *behavior* (*i.e.*, business logic independent), (5) platforms, and (6) programming languages.

In fact, as specified by the Reactive Manifesto [2], event-driven design is a pre-requisite for building systems that are *reactive*,*i.e.* readily responsive to incoming data, user interaction events, failures and load variations- traits which are desirable of critical systems. Moreover, asynchronous event-based architectures unify scaling up (e.g., via multiple cores) and scaling out (e.g., via distributed compute nodes) while deferring the choice of the scalability mechanism at deployment-time without hiding *the network* from the programming model. Hence, the asynchronous and event-driven programming model offered by DDS makes it particularly well-suited for demanding systems.

However, the data processing aspects, which are local to the individual stages of a distributed dataflow, are often not implemented as a dataflow due to lack of sufficient composability and generality in the application programming interface (API) of the pub/sub middleware. DDS offers various ways to receive data such as, listener callbacks for push-based notification, read/take functions for polling, waitset and read-condition to receive data from several entities at a time, and query-conditions to enable application-specific filtering and demultiplexing. These primitives, however, are designed for data and meta-data *delivery* as opposed to *processing*. Further, the lack of proper abstractions forces programmers to develop event-driven applications using the observer pattern– disadvantages of which are well documented [90].

To that end we have investigated the use of Functional Reactive Programming (FRP) for DDS applications [82]. FRP has emerged [136] as a promising new way to create scalable reactive applications and has already shown its potential in a number of domains including robotics [108, 146], animation [39], HD video streaming [1], and responsive user interfaces [92]. These domains are reactive because they require interaction with a wide range of inputs ranging from keyboards to machinery. FRP is a declarative approach to system development wherein program specification amounts to "what" as opposed "how". Such a program description can be viewed as a data-flow [30] system where the state and control flow are hidden from the programmers. FRP offers high-level abstractions that avoid the verbosity that is commonly observed in callback-based techniques. Furthermore, FRP avoids shared mutable state at the application-level, which is instrumental for multicore scalability.

3.4 Middleware Specialization

High confidence software, which includes the middleware platforms, is a crucial design consideration for cyber physical systems. Developing and deploying high confidence software for cyber physical systems is fraught with multiple challenges. Addressing these challenges and realizing the goals of CPS has been a major research focus for me since my promotion and tenure.

Due to the varying constraints of both the operating environment of CPS and constraints on the resources, CPS applications often do not require all the features provided by contemporary, general-purpose middleware. A lack of support for selective use of features forces applications to incur the overhead of linking in all the capabilities, which degrades QoS and increases memory footprint. Second, not only does general-purpose middleware lack *out-of-the-box* support for diverse and rich domain-specific properties (*e.g.*, a custom fault masking and recovery capability), but also lack modular extensibility for both domain-specific and -independent features.

A promising approach to address this problem is to *specialize* general-purpose middleware (*i.e.*, prune unwanted features, add and customize the necessary ones). The research I conducted as part of my NSF CAREER research involved (a) identifying and exploiting the algebraic structure of middleware that helps map the specialization problem into a feature-oriented software development problem; (b) developing a new theory for feature composition, refactoring, and interactions across the lifecycle stages of applications; and (c) automating these specializations through well-defined processes.

The **Intellectual Merits** of this research included novel software engineering approaches to specializing middleware [33, 35, 34, 142], model-based techniques [79, 24, 5, 132, 7], fault tolerance algorithms [140, 141, 32] and resource management algorithms [8, 4, 42, 149, 9, 10].

The ***broader impact*** from this research included (a) stimulating new directions in middleware design with use cases in virtualization environments; (b) documentation of *specialization patterns*, which help to reduce software maintenance and configuration challenges in product lines; (c) enhanced developer productivity and system correctness; and (d) education and technology transfer. We created an ***open source*** tool suite called *GAMMA* (Generative Aspects for the Manipulation of Middleware Architectures).

3.5 Deployment and Configuration Heuristics

Determining how to deploy software to hardware in CPS is a challenging problem. Developers must ensure that each software component is provided sufficient processing time to meet any real-time scheduling constraints. Resource constraints, such as total available memory on each processor, must also be respected. Finally, components can have complex placement or co-location constraints, such as requiring specific software components to be deployed to processors due to physical constraints imposed by the domain.

An NSF CNS/SHS award supported this research. The Intellectual Merit of this work lies in the investigations of formalisms and techniques for effective deployment and configuration of distributed, real-time and embedded systems. To that end we have devised mechanisms both as mathematical formulations as well as middleware artifacts, such as optimizations for runtime deployment and configuration. One such approach resulted in new hybrid algorithmic techniques for deployment of CPS. It comprises new bin-packing based deployment algorithms that accept co-location, scheduling, and resource constraints. It contained hybrid bin-packing/evolutionary algorithms for handling network constraints and non-linear optimization. The complete research resulted in a number of publications. [15, 122, 121, 131, 11, 120, 110, 102, 103, 144, 38, 104, 109, 12, 24, 8, 4, 114, 101].

The ***broader impact*** from this research include less expensive and correct deployment of and energy efficient CPS. By packing software more tightly onto hardware, fewer hardware resources can be used. For example, roughly four pounds of cooling, power supply, and other supporting hardware are needed for each pound of processing hardware in a plane. Reducing hardware not only decreases CPS cost, but also can facilitate increased ranges for planes/cars, decreased fuel and power consumption, and reduced waste.

4 Foundational Research Activities at Vanderbilt University (2002-2009)

My research activities at Vanderbilt University prior to obtaining tenure first started as a research scientist in the Institute for Software Integrated Systems (ISIS) in Jan 2002 and subsequently as a tenure-track assistant professor (starting Sept 2003)in the Department of Electrical Engineering and Computer Science has investigated solutions to address the deployment and configuration, and QoS management challenges in component-based DRE systems.

4.1 Contributions to Science and Broader Impact: Model-driven Middleware

To better enable the use of component middleware and rapid application composition with assured QoS in the context of DRE systems, my research has formulated the *Model-driven Middleware* paradigm, which combines the strengths of model-driven engineering (MDE) [125] with that of QoS-enabled component middleware [147]. The key research contributions and the challenges they resolve are described below:

1. Technology for effectively composing DRE systems from components. QoS-enabled component middleware enables application developers to develop individual components that can be composed together into *assemblies* that form complete DRE systems. Although this approach supports the use of "plug and play" components in DRE systems, system integrators face the daunting task of composing the right set of compatible components that will deliver the desired semantics and QoS to applications that execute in large-scale systems.

2. Technology for configuring component middleware. In QoS-enabled component middleware frameworks, application components and the underlying component middleware services can have a large number of attributes and parameters that can be configured at various stages of the development lifecycle, such as (1) *during component development*, where default values for these attributes could be specified, (2) *during application integration*, where component defaults could be overridden with domain specific defaults, and (3) *during application deployment*, where domain specific defaults are overridden based on the actual capabilities of the target system. It is hard, however, to manually ensure that all these parameters are semantically consistent throughout a large-scale DRE system.

3. Technology for automated deployment of DRE systems on heterogeneous target platforms. The component assemblies described above must be deployed in the distributed target environment before applications can start to run. DRE system integrators must therefore perform the complex task of mapping the individual components/assemblies onto specific nodes of the target environment. This mapping process involves ensuring semantic compatibility between the requirements of the individual components, and the capabilities of the nodes of the target environment.

4. Technology for performance evaluation for DRE systems. It is critical that the component assemblies, their configurations and deployment decisions be validated for conformance to their QoS requirements. We realized that there was a dearth of tools and techniques that enabled automated performance evaluation as well as provided design-time estimates of the expected performance of the overall system.

5. Technology for survivable DRE systems. It is essential that the operationalized DRE system continue to deliver the QoS despite fluctuations in resource availabilities and failures. This requires adaptive solutions that maintain both real-time (the performance aspect) and fault-tolerance (the availability aspect) of the DRE system.

4.2 Research Artifacts

The research artifacts stemming from my foundational research include an open source MDE tool suite called CoSMIC and a middleware infrastructure that supports both real-time and fault-tolerance.

4.2.1 CoSMIC Tool Suite

The *Component Synthesis using Model Integrated Computing* (CoSMIC) [46] tool suite is an integrated collection of model-driven environment (MDE) tools that address the key lifecycle challenges of middleware and applications in DRE systems. CoSMIC comprises the following capabilities:

I. Modeling Tools for Deployment and Configuration: CoSMIC is a collection of domain-specific modeling languages (DSMLs) [85, 18, 61], and model interpreters and transformations [77, 81, 80, 78] that enforce the principle of "correct-by-construction.". CoSMIC [46, 57, 22, 19, 20], provides a model-driven, generative programming approach [47] to resolve tangled QoS concerns [21] of middleware-based real-time systems [57]. This includes using metamodeling [137] and domain-specific modeling languages (DSMLs) [61] to separate the application assembly, packaging, configuration and deployment concerns [143]. Generative technologies [31] are used to weave [138, 16, 139] these concerns into middleware platforms in a scalable manner [60].

II. Modeling Tools for Design-time Performance Validation: This capability can be further classified along the following three dimensions:

(a) Continuous QoS validation, which focuses on emulating the business logic of application components so that end-to-end behavior can be emulated and validated continuously throughout the application development, integration and deployment phase [71, 70, 68, 69]. Application behavior is modeled using the Component behavior Modeling Language (CBML), which is based on the mathematical formalism of Input/Output Automata [89], and the workloads are modeling using the Workload Modeling Language (WML).

(b) Validating the structure and configurations of the middleware infrastructure, where my team and I have developed model-based performance analysis solutions [58, 83, 59, 115, 116] that leverage Stochastic Rewards Nets (SRN) [93, 67] solvers to conduct automated performance analysis of different software patterns at a platform-independent level. We developed a DSML in CoSMIC called the Pattern-Oriented Software Architecture Modeling Language (POSAML) [75] to model the structure and features of middleware stacks in the form of patterns-based [130, 44] building blocks.

(c) Continuous QoS assurance via benchmarking and model checking: In [85, 86, 150] along with my collaborators I have demonstrated how benchmarking code generation techniques within CoSMIC can be applied to discover the dominant configuration options in middleware that impact system QoS. I have also used model checking [119, 73] as a means to verify the correctness of generated middleware configurations [76, 81].

4.2.2 FLARe and CORFU Middleware

My other significant research contribution pertains to the design and implementation of a real-time and fault-tolerant middleware. FLARe (Fault-tolerant Load-aware and Adaptive middlewaRe) [14, 17] provides a load-aware adaptive middleware that minimizes client response times while maximizing the system availability despite resource unavailability and failures. CORFU (COmponent Replication using Failover Units) [148, 72] leverages FLARe and makes it available to components that are replicated and treated logically as failover units.

5 Industrial Research (1998 – 2002)

This section outlines my research contributions as a member of technical staff (MTS) from July 1998 to Jan 2002 at Bell Laboratories, Lucent Technologies (now Nokia Bell Labs) in New Jersey.

5.1 Contributions to Science and Broader Impact: Fault Tolerant Middleware Standard

My research contributions towards addressing the need for survivable and highly available systems explored novel mechanisms to provide these assurances to applications via reusable middleware services. This research contributed to the Fault-tolerant CORBA standard.

Additional research contributions I made in the realm of customer relationship management (CRM) led to the development of efficient agent allocation algorithms that help improve the performance and reduce the costs of call center activities. This research resulted in a patent application. Moreover, it is conceivable that such algorithms can be useful in 911 emergency management call centers too.

5.2 Research Artifacts: DOORS FT CORBA

My research contributions included a prototype implementation of Fault Tolerant CORBA called DOORS that included performance optimizations [94, 95], and its standardization [96] activities within the Object Management Group (OMG). I was also involved with designing and developing fault tolerant services used in third-generation wireless call processing elements that were based on the UMTS standard. Additionally, a patent has been filed for my contributions to the customer-agent allocation algorithms in the context of networked call centers and customer relationship management.

6 Doctoral Research (1992-1998)

My research focus as a doctoral student at Washington University in St. Louis was on a class of systems called the distributed, real-time and embedded (DRE) systems found in numerous domains, such as avionics mission computing, shipboard computing and industrial automation. My research focused on designing and implementing robust and optimized middleware hosting platforms for DRE systems.

6.1 Contributions to Science and Broader Impact: QoS-enabled Middleware

My doctoral research led to a key enabling technology to realize the goals of a service-oriented cyber infrastructure to host DRE systems. It realized the notion of *performance-tuned middleware* [124]. Specifically, my dissertation focused on devising algorithms and optimization techniques that substantially improved the performance and predictability of real-time middleware.

The research contributions included: (1) efficient *protocol engines* [54] based on interpretive (de)marshaling mechanisms that minimized footprint while enabling performance that rivaled compiled (de)marshaling techniques, (2) efficient *request demultiplexers* [52] that combined perfect hashing and active demultiplexing for constant time dispatching of messages to object implementations thereby improving predictability and scalability, and (3) generative programming [31] via *interface definition language compilers* [55] that used aspect-oriented generative techniques [43] to weave interpreted and compiled stubs together to achieve optimal trade-offs between footprint and performance.

A notable outcome of my doctoral research has been the documentation and codification of *optimization principle patterns* [56, 118], such as optimizing for the common case; eliminating gratuitous waste; replacing general purpose methods with specialized, efficient ones; precomputing values, if possible; storing redundant state to speed up expensive operations; passing information between layers; optimizing for the processor cache; and optimizing demultiplexing strategies.

6.2 Research Artifacts: The TAO ORB

My doctoral research resulted in several novel middleware benchmarking [48, 49, 51, 52] and middleware optimization techniques [54, 50, 53, 128, 55, 56, 117], which have been used to guide the implementation of a high-performance, real-time CORBA [97] Object Request Broker (ORB) [98] called TAO [126, 127, 129], which also substantially influenced the Real-time CORBA specification [97].

7 Research Management Plan

This section describes briefly how I plan to execute my research tasks to achieve both near- and longer-term objectives.

7.1 Expected Doctoral Advisee Contributions

In the following I outline the expected contributions of my current doctoral student advisees and co-advisees leading to the realization of the stated goals of my research.

1. **Shunxing Bao,** PhD expected Spring 2018; focusing on performance optimizations to Apache Hadoop ecosystem for medical image processing applications.
2. **Yogesh Barve,** PhD expected Fall 2018; focusing on reliability in high performance, distributed simulations.
3. **Anirban Bhattacharjee,** PhD expected Fall 2018; focusing on model-driven orchestration of Big Data cloud-based systems.
4. **Travis Brummett,** is a new PhD student in my group.
5. **Davut Disci,** PhD expected Spring 2019; starting to focus on Big Data storage management and technologies for STEM education.
6. **Shweta Khare,** PhD expected Fall 2018; focusing on blending real-time stream processing with data-centric publish/subscribe.
7. **Shashank Shekhar,** PhD expected Spring 2018; focusing on dynamic resource management for cloudlet/edge computing.

7.2 Research Collaborations

I am collaborating with a number of internal and external collaborators. In the following I summarize how I expect my collaborations to aid in my ongoing and future research endeavors.

1. **Drs. Xenofon Koutsoukos and Yevgeniy Vorobeychik (ISIS/EECS, VU),** with whom I am teaming up to conduct follow-on research in the DDDAS area and awaiting a decision on a follow-on grant for the DDDAS program.
2. **Drs. Abhishek Dubey (ISIS/EECS, VU),** with whom I am collaborating to further advance our work in Edge/Cloud computing. We have recently won a DURIP equipment award and will be setting up a testbed for our investigations involving joint work in Smart Cities.
3. **Dr. Bennett Landman (EECS, VU),** with whom I have teamed up for over an year and am co-advising a student, who is investigating performance optimizations in the Apache Hadoop ecosystem for medical image processing applications.
4. **Dr. Gautam Biswas (EECS, VU),** with whom I have been working for the past few years on technologies for STEM education. Currently we are funded by the NSF US Ignite program where we are exploring the use of software defined networking technologies to use cloudlets and the edge for low latency collaborative STEM learning.
5. **Drs. Abhishek Dubey, Gautam Biswas, Douglas Schmidt, Jules White (ISIS/EECS, VU), Dr. Hiba Baroud (CEE, VU) and several other VU colleagues outside of the School of Engineering,** with whom I have teamed up to conduct research on Smart Cities, which is funded by the Vanderbilt TIPS program.
6. **Dr. Shivakumar Sastry (ECE, Univ of Akron),** with whom I have been collaborating for the past few years in addressing a variety of problems related to advanced manufacturing.

7. **Dr. Sumant Tambe (LinkedIn) and Dr. Kyoungho An (RTI),** with whom I have been collaborating to investigate ideas in real-time stream processing and publish/subscribe systems.
8. **Dr. Akram Hakiri (Tunisia),** with whom I have been collaborating for the past few years on topics related to software-defined networking.
9. **Dr. Takayuki Kuroda (NEC, Japan),** with whom I have been collaborating since his sabbatical in my lab during 2014-15 academic year on configuration automation for enterprise systems.
10. **Dr. Yogesh Simmhan (IISc, Bengaluru, India),** with whom I had an initial kickoff meeting to forge collaborative efforts given substantial synergies in our work.

References

[1] Reactive Programming at Netflix. http://techblog.netflix.com/2013/01/reactive-programming-at-netflix.html, 2013. 3.3

[2] The Reactive Manifesto. http://www.reactivemanifesto.org, 2013. 3.3

[3] A. Alamri, W. S. Ansari, M. M. Hassan, M. S. Hossain, A. Alelaiwi, and M. A. Hossain. A Survey on Sensor-cloud: Architecture, Applications, and Approaches. *International Journal of Distributed Sensor Networks*, 2013, 2013. 3.1.5

[4] K. An, F. Caglar, S. Shekhar, and A. Gokhale. Automated Placement of Virtual Machine Replicas to Support Reliable Distributed Real-time and Embedded Systems in the Cloud. In *International Workshop on Real-time and Distributed Computing in Emerging Applications (REACTION), 33rd IEEE Real-time Systems Symposium (RTSS '12)*, San Juan, Puerto Rico, USA, Dec. 2012. IEEE. 3.4, 3.5

[5] K. An and A. Gokhale. Model-driven Performance Analysis and Deployment Planning for Real-time Stream Processing. In *Proceedings of the Work-in-Progress Session of the 19th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '13)*, pages 21–24, Philadelphia, PA, USA, Apr. 2013. IEEE. 3.4

[6] K. An, S. Khare, A. Gokhale, and A. Hakiri. Experience Paper: An Autonomous and Dynamic Coordination and Discovery Service for Wide-Area Peer-to-peer Publish/Subscribe. In *11th ACM International Conference on Distributed and Event-Based Systems (DEBS)*, pages 239–248, Barcelona, Spain, June 2017. ACM. 2

[7] K. An, T. Kuroda, A. Gokhale, S. Tambe, and A. Sorbini. Model-driven Generative Framework for Automated OMG DDS Performance Testing in the Cloud. In *12th International Conference on Generative Programming: Concepts & Experiences (GPCE'13)*, pages 179–182, Indianapolis, IN, USA, Oct. 2013. ACM. 3.4

[8] K. An, S. Pradhan, F. Caglar, and A. Gokhale. A Publish/Subscribe Middleware for Dependable and Real-time Resource Monitoring in the Cloud. In *Secure and Dependable Middleware for Cloud Monitoring and Management (SDMCMM '12) Workshop at ACM/USENIX/IFIP Middleware 2012*, Montreal, Canada, Dec. 2012. ACM. 3.4, 3.5

[9] K. An, S. Shekhar, F. Caglar, A. Gokhale, and S. Sastry. A Cloud Middleware for Assuring Performance and High Availability of Soft Real-time Applications. *Elsevier Journal of Systems Architecture (JSA)*, 60(9):757–769, Dec. 2014. 3.4

[10] K. An, S. Tambe, P. Pazandak, G. Pardo-Castello, A. Gokhale, and D. C. Schmidt. Content-based Filtering Discovery Protocol (CFDP): Scalable and Efficient OMG DDS Discovery Protocol. In *8th ACM International Conference on Distributed Event-based Systems (DEBS)*, pages 130–141, Mumbai, India, May 2014. IEEE. 3.4

[11] K. An, A. Trewyn, A. Gokhale, and S. Sastry. Model-driven Performance Analysis of Reconfigurable Conveyor Systems used in Material Handling Applications. In *Second IEEE/ACM International Conference on Cyber Physical Systems (ICCPS 2011)*, pages 141–150, Chicago, IL, USA, Apr. 2011. IEEE. 3.5

[12] K. An, A. Trewyn, A. Gokhale, and S. Sastry. *Formal and Practical Aspects of Domain-specific Languages: Recent Developments*, chapter Design and Transformation of a Domain-specific Language for Reconfigurable Conveyor Systems, Chapt 19, pages 551–569. IGI Global, July 2013. 3.5

[13] L. Atzori, A. Iera, and G. Morabito. The Internet of Things: A Survey. *Computer networks*, 54(15):2787–2805, 2010. 3.1.5

[14] J. Balasubramanian, A. Gokhale, D. C. Schmidt, and N. Wang. Towards Middleware for Fault-tolerance in Distributed Real-time and Embedded Systems. In *Proceedings of the 8th International Conference on Distributed Applications and Interoperable Systems (DAIS 2008)*, pages 72–85, Oslo, Norway, June 2008. IFIP. 4.2.2

[15] J. Balasubramanian, A. Gokhale, F. Wolf, A. Dubey, C. Lu, C. Gill, and D. C. Schmidt. Resource-Aware Deployment and Configuration of Fault-tolerant Real-time Systems. In *Proceedings of the 16th IEEE Real-time and Embedded Technology and Applications Symposium (RTAS '10)*, pages 69–78, Stockholm, Sweden, Apr. 2010. 3.5

[16] J. Balasubramanian, S. Tambe, B. Dasarathy, S. Gadgil, F. Porter, A. Gokhale, and D. C. Schmidt. Netqope: A model-driven network qos provisioning engine for distributed real-time and embedded systems. In *RTAS' 08: Proceedings of the 14th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 113–122, Los Alamitos, CA, USA, 2008. IEEE Computer Society. 4.2.1

[17] J. Balasubramanian, S. Tambe, C. Lu, A. Gokhale, C. Gill, and D. C. Schmidt. Adaptive Failover for Real-time Middleware with Passive Replication. In *Proceedings of the 15th Real-time and Embedded Applications Symposium (RTAS '09)*, pages 118–127, San Francisco, CA, Apr. 2009. 4.2.2

[18] K. Balasubramanian, J. Balasubramanian, J. Parsons, A. Gokhale, and D. C. Schmidt. A Platform-Independent Component Modeling Language for Distributed Real-Time and Embedded Systems. In *RTAS '05: Proceedings of the 11th IEEE Real Time on Embedded Technology and Applications Symposium*, pages 190–199, Washington, DC, USA, 2005. IEEE Computer Society. 4.2.1

[19] K. Balasubramanian, J. Balasubramanian, J. Parsons, A. Gokhale, and D. C. Schmidt. A Platform-Independent Component Modeling Language for Distributed Real-time and Embedded Systems. *Journal of Computer Systems Science*, 73(2):171–185, 2007. 4.2.1

[20] K. Balasubramanian, A. Gokhale, J. Sztipanovits, G. Karsai, and S. Neema. Developing Applications Using Model-Driven Design Environments. *IEEE Computer*, 39(2):33–40, Feb. 2006. 4.2.1

[21] K. Balasubramanian, A. S. Gokhale, Y. Lin, J. Zhang, and J. Gray. Weaving Deployment Aspects into Domain-specific Models. *International Journal of Software Engineering and Knowledge Engineering*, 16(3):403–424, 2006. 4.2.1

[22] K. Balasubramanian, A. S. Krishna, E. Turkay, J. Balasubramanian, J. Parsons, A. Gokhale, and D. C. Schmidt. Applying Model-Driven Development to Distributed Real-time and Embedded Avionics Systems. *International Journal of Embedded Systems: Special Issue on the Design and Verification of Real-Time Embedded Software*, 2:142–155, 2006. 4.2.1

[23] S. Bao, A. Plassard, B. A. Landman, and A. Gokhale. Cloud Engineering Principles and Technology Enablers for Medical Image Processing-as-a-Service. In *IEEE International Conference on Cloud Engineering (IC2E)*, pages 127–137, Vancouver, Canada, Apr. 2017. IEEE. 2

[24] F. Caglar, K. An, A. Gokhale, and T. Levendovszky. Transitioning to the Cloud? A Model-driven Analysis and Automated Deployment Capability for Cloud Services. In *1st International Workshop on Model-Driven Engineering for High Performance and CLoud computing (MDHPCL) at MODELS 2012*, Innsbruck, Austria, Oct. 2012. ACM/IEEE. 3.4, 3.5

[25] F. Caglar and A. Gokhale. iOverbook: Managing Cloud-based Soft Real-time Applications in a Resource-Overbooked Data Center. In *The 7th IEEE International Conference on Cloud Computing (CLOUD' 14)*, pages 538–545, Anchorage, AL, USA, June 2014. IEEE. 2, 3.1.2

[26] F. Caglar, S. Shekhar, and A. Gokhale. iPlace: An Intelligent and Tunable Power- and Performance-Aware Virtual Machine Placement Technique for Cloud-based Real-time Applications. In *17th IEEE Computer Society Symposium on Object/component/service-oriented real-time distributed Computing Technology (ISORC '14).*, pages 48–55, Reno, NV, USA, June 2014. IEEE. 3.1.4

[27] F. Caglar, S. Shekhar, and A. Gokhale. iTune: Engineering the Performance of Xen Hypervisor via Autonomous and Dynamic Scheduler Reconfiguration. *IEEE Transactions on Services Computing (TSC)*, PP(99), Mar. 2016. 2, 3.1.1

[28] F. Caglar, S. Shekhar, A. Gokhale, S. Basu, T. Rafi, J. Kinnebrew, and G. Biswas. Cloud-hosted Simulation-as-a-Service for High School STEM Education. *Elsevier Simulation Modelling Practice and Theory: Special Issue on Cloud Simulation*, 58(2):255–273, Nov. 2015. 2

[29] F. Caglar, S. Shekhar, A. Gokhale, and X. Koutsoukos. An Intelligent, Performance Interference-aware Resource Management Scheme for IoT Cloud Backends. In *1st IEEE International Conference on Internet-of-Things: Design and Implementation*, pages 95–105, Berlin, Germany, Apr. 2016. IEEE. 2, 3.1.3

[30] G. H. Cooper and S. Krishnamurthi. Embedding Dynamic Dataflow in a Call-by-value Language. In *Programming Languages and Systems*, pages 294–308. Springer, 2006. 3.3

[31] K. Czarnecki and U. W. Eisenecker. *Generative Programming: Methods, Tools, and Applications*. Addison-Wesley, Reading, Massachusetts, 2000. 4.2.1, 6.1

[32] A. Dabholkar, A. Dubey, A. Gokhale, G. Karsai, and N. Mahadevan. Reliable Distributed Real-Time and Embedded Systems through Safe Middleware Adaptation. In *31st IEEE International Symposium on Reliable Distributed Systems (SRDS '12)*, pages 362–371, Irvine, CA, USA, Oct. 2012. IEEE. 3.4

[33] A. Dabholkar and A. Gokhale. Middleware specialization for product-lines using feature- oriented reverse engineering. In S. Latifi, editor, *Seventh International Conference on Information Technology: New Generations (ITNG 2010)*, pages 696–701, Las Vegas, Nevada, USA, April 12-14 2010. IEEE Computer Society. 3.4

[34] A. Dabholkar and A. Gokhale. A Generative Middleware Specialization Process for Distributed Real-time and Embedded Systems. In *14th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC 2011)*, pages 197–204, Newport Beach, CA, USA, Mar. 2011. IEEE. 3.4

[35] A. Dabholkar and A. Gokhale. FORMS: Feature-Oriented Reverse Engineering-based Middleware Specialization for Product-Lines. *Journal of Software (JSW) - Special Issue on Recent Advances in Middleware and Network Applications*, 6(4):519–527, 2011. 3.4

[36] A. Dukeman, F. Caglar, S. Shekhar, J. Kinnebrew, G. Biswas, D. Fisher, and A. Gokhale. Teaching Computational Thinking Skills in C3STEM with Traffic Simulation. In A. Holzinger and G. Pasi, editors, *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, volume 7947 of *Lecture Notes in Computer Science*, pages 350–357. Springer Berlin Heidelberg, 2013. 2

[37] A. Dukeman, L. Hou, S. Shekhar, F. Caglar, J. Kinnebrew, G. Biswas, A. Gokhale, and D. Fisher. Modeling Student Program Evolution in STEM Disciplines. In *Poster paper at the 121st ASEE Annual Conference, K-12 and Pre-Engineering Track*. ASEE, Indianapolis, IN, USA, June 2014. 2

[38] J. Ehrenfeld, A. Gokhale, X. Koutsoukos, and D. C. Schmidt. A Closed Loop Control Architecture to Maintain Patient Normothermia During Perioperative Periods. In *Work-in-Progress Session of ACM/IEEE 3rd International Conference on Cyber-Physical Systems (ICCPS '12)*, page 217, Beijing, China, Apr. 2012. ACM/IEEE. 3.5

[39] C. Elliott and P. Hudak. Functional Reactive Animation. In *ACM SIGPLAN Notices*, volume 32, pages 263–273. ACM, 1997. 3.3

[40] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. *ACM Comput. Surv.* 3.3

[41] P. C. Evans and M. Annunziata. Industrial Internet: Pushing the Boundaries of Minds and Machines. *General Electric*, page 21, 2012. 2

[42] K. A. Faruk Caglar, Shashank Shekhar and A. Gokhale. Intelligent Power- and Performance-aware Tradeoffs for Multicore Servers in Cloud Data Centers. In *Proceedings of the Work-in-Progress Session of the 4th ACM/IEEE International Conference on Cyber Physical Systems (ICCPS' 13)*, Philadelphia, PA, USA, Apr. 2013. IEEE/ACM. 3.4

[43] R. Filman, T. Elrad, M. Aksit, and S. Clarke, editors. *Aspect-Oriented Software Development*. Addison-Wesley, Reading, Massachusetts, 2003. 6.1

[44] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA, 1995. 4.2.1

[45] GE Software. The Case for an Industrial Big Data Platform: Laying the Groundwork for the New Industrial Age. `http://www.ge.com/digital/sites/default/files/Industrial_Big_Data_Platform.pdf`, 2013. 2

[46] A. Gokhale, K. Balasubramanian, J. Balasubramanian, A. S. Krishna, G. T. Edwards, G. Deng, E. Turkay, J. Parsons, and D. C. Schmidt. Model Driven Middleware: A New Paradigm for Deploying and Provisioning Distributed Real-time and Embedded Applications. *The Journal of Science of Computer Programming: Special Issue on Foundations and Applications of Model Driven Architecture (MDA)*, 73(1):39–58, 2008. 4.2.1, 4.2.1

[47] A. Gokhale, B. Natarajan, D. C. Schmidt, A. Nechypurenko, J. Gray, N. Wang, S. Neema, T. Bapty, and J. Parsons. CoSMIC: An MDA Generative Tool for Distributed Real-time and Embedded Component Middleware and Applications. In *Proceedings of the OOPSLA 2002 Workshop on Generative Techniques in the Context of Model Driven Architecture*, Seattle, WA, Nov. 2002. ACM. 4.2.1

[48] A. Gokhale and D. C. Schmidt. Measuring the Performance of Communication Middleware on High-Speed Networks. In *Proceedings of SIGCOMM '96*, pages 306–317, Stanford, CA, Aug. 1996. ACM. 6.2

[49] A. Gokhale and D. C. Schmidt. The Performance of the CORBA Dynamic Invocation Interface and Dynamic Skeleton Interface over High-Speed ATM Networks. In *Proceedings of GLOBECOM '96*, pages 50–56, London, England, Nov. 1996. IEEE. 6.2

[50] A. Gokhale and D. C. Schmidt. Design Principles and Optimizations for High-performance ORBs. In 12^{th} OOPSLA Conference, poster session, Atlanta, Georgia, Oct. 1997. ACM. 6.2

[51] A. Gokhale and D. C. Schmidt. Evaluating Latency and Scalability of CORBA Over High-Speed ATM Networks. In *Proceedings of the International Conference on Distributed Computing Systems*, Baltimore, Maryland, May 1997. IEEE. 6.2

[52] A. Gokhale and D. C. Schmidt. Evaluating the Performance of Demultiplexing Strategies for Real-time CORBA. In *Proceedings of GLOBECOM '97*, Phoenix, AZ, Nov. 1997. IEEE. 6.1, 6.2

[53] A. Gokhale and D. C. Schmidt. Measuring and Optimizing CORBA Latency and Scalability Over High-speed Networks. *Transactions on Computing*, 47(4), 1998. 6.2

[54] A. Gokhale and D. C. Schmidt. Principles for Optimizing CORBA Internet Inter-ORB Protocol Performance. In *Hawaiian International Conference on System Sciences*, Hawaii, USA, Jan. 1998. 6.1, 6.2

[55] A. Gokhale and D. C. Schmidt. Optimizing a CORBA IIOP Protocol Engine for Minimal Footprint Multimedia Systems. *Journal on Selected Areas in Communications Special Issue on Service Enabling Platforms for Networked Multimedia Systems*, 17(9), Sept. 1999. 6.1, 6.2

[56] A. Gokhale and D. C. Schmidt. Techniques for Optimizing CORBA Middleware for Distributed Embedded Systems. In *Proceedings of INFOCOM '99*, Mar. 1999. 6.1, 6.2

[57] A. Gokhale, D. C. Schmidt, B. Natarajan, J. Gray, and N. Wang. Model Driven Middleware. In Q. Mahmoud, editor, *Middleware for Communications*, pages 163–187. Wiley and Sons, New York, 2004. 4.2.1

[58] S. Gokhale, A. Gokhale, and J. Gray. Response Time Analysis of Middleware Event Demultiplexing Pattern for Network Services. In *Proceedings of IEEE Globecom, Advances for Networks and Internet Track*, St. Louis, MO, December 2005. 4.2.1

[59] S. Gokhale, P. Vandal, U. Praphamontripong, A. Gokhale, and J. Gray. Performance Analysis of the Reactor Pattern in Network Services. In *Proceedings of the 5th International Workshop on Performance Modeling, Evaluation and Optimization of Parallel and Distributed Systems (PMEO-PDS 2006)*, Rhodes Island, Greece, April 2006. 4.2.1

[60] J. Gray, Y. Lin, J. Zhang, S. Nordstrom, A. Gokhale, S. Neema, and S. Gokhale. Replicators: Transformations to Address Model Scalability. In *Lecture Notes in Computer Science: Proceedings of 8th International Conference Model Driven Engineering Languages and Systems (MoDELS 2005)*, pages 295–308, Montego Bay, Jamaica, Nov. 2005. Springer Verlag. 4.2.1

[61] J. Gray, J. Tolvanen, S. Kelly, A. Gokhale, S. Neema, and J. Sprinkle. Domain-Specific Modeling. In *CRC Handbook on Dynamic System Modeling, (Paul Fishwick, ed.)*, pages 7.1–7.20. CRC Press, May 2007. 4.2.1

[62] A. Hakiri, P. Berthou, and A. Gokhale. Publish/Subscribe-enabled Software Defined Networking for Efficient and Scalable IoT Communications. *IEEE Communications Magazine, Communications Standards Supplement*, 53(9):48–54, Sept. 2015. 2, 3.2

[63] A. Hakiri and A. Gokhale. Data-centric Publish/Subscribe Routing Middleware for Realizing Proactive Overlay Software-defined Networking. In *ACM International Conference on Distributed and Event-based Systems (DEBS)*, pages 246–257, Irvine, CA, USA, June 2016. ACM. 2, 3.2

[64] A. Hakiri and A. Gokhale. Rethinking the Design of LR-WPAN IoT Systems with Software-Defined Networking. In *International Conference on Distributed Computing in Sensor Networks (DCOSS)*, pages 238–243, Washington, DC, USA, May 2016. IEEE. 3.2

[65] A. Hakiri, A. Gokhale, P. Berthou, D. Schmidt, and T. Gayraud. Software Defined Networking: Challenges and Research Opportunities for Future Internet. *Elsevier Journal of Computer Networks (COMNET)*, 75(A):453–471, Dec. 2014. 2, 3.2

[66] A. Hakiri, P. Patil, and A. Gokhale. SDN-enabled Wireless Fog Network Management. In *Submitted to IEEE Conference on Communications (ICC)*, Paris, France, May 2017. IEEE. 3.2

[67] B. R. Haverkort and K. S. Trivedi. "Specification and generation of Markov reward models". *Discrete–Event Dynamic Systems: Theory and Applications*, 3:219–247, 1993. 4.2.1

[68] J. H. Hill and A. Gokhale. Model-driven Specification of Component-based Distributed Real-time and Embedded Systems for Verification of Systemic QoS Properties. In *Proceeding of the Workshop on Parallel, Distributed, and Real-Time Systems (WPDRTS '08)*, Miami, FL, Apr. 2008. 4.2.1

[69] J. H. Hill and A. Gokhale. Towards Improving End-to-End Performance of Distributed Real-time and Embedded Systems using Baseline Profiles. *Software Engineering Research, Management and Applications (SERA 08), Special Issue of Springer Journal of Studies in Computational Intelligence*, 150(14):43–57, Aug. 2008. 4.2.1

[70] J. H. Hill and A. S. Gokhale. Model-driven engineering for early qos validation of component-based software systems. *JSW*, 2(3):9–18, 2007. 4.2.1

[71] J. H. Hill, S. Tambe, and A. Gokhale. Model-driven Engineering for Development-time QoS Validation of Component-based Software Systems. In *Proceedings of 14th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 07)*, pages 307–316, Tucson, AZ, Mar 2007. 4.2.1

[72] J. Hoffert, A. Gokhale, and D. Schmidt. Evaluating Transport Protocols for Real-time Event Stream Processing Middleware and Applications. In *Proceedings of the 11th International Symposium on Distributed Objects, Middleware, and Applications (DOA '09)*, Vilamoura, Algarve-Portugal, Nov. 2009. 4.2.2

[73] G. J. Holtzman. The Model Checker SPIN. *IEEE Transactions on Software Engineering*, 23(5):279–295, May 1997. 4.2.1

[74] I. C. Inc. What happens in an internet minute? `http://www.intel.com/content/www/us/en/communications/internet-minute-infographic.html`, 2014. 3.1

[75] D. Kaul, A. Kogekar, A. Gokhale, J. Gray, and S. Gokhale. Managing Variability in Middleware Provisioning Using Visual Modeling Languages. In *Proceedings of the Hawaii International Conference on System Sciences HICSS-40 (2007), Visual Interactions in Software Artifacts Minitrack, Software Technology Track*, Big Island, Hawaii, Jan 2007. 4.2.1

[76] A. Kavimandan, K. Balasubramanian, N. Shankaran, A. Gokhale, and D. C. Schmidt. Quicker: A model-driven qos mapping tool for qos-enabled component middleware. In *ISORC '07: Proceedings of the 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing*, pages 62–70, Washington, DC, USA, 2007. IEEE Computer Society. 4.2.1

[77] A. Kavimandan and A. Gokhale. Automated Middleware QoS Configuration Techniques using Model Transformations. In *Proceedings of the* 14th *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2008)*, pages 93–102, St. Louis, MO, USA, Apr. 2008. 4.2.1

[78] A. Kavimandan and A. Gokhale. A Model-transformation Approach to Improving the Quality of Software Architectures for Distributed Real-time and Embedded Systems. In *Proceedings of the* 5th *International Conference on the Quality of Software Architectures (QoSA '09)*, pages 18–35, East Stroudsberg, PA, June 2009. ACM. 4.2.1

[79] A. Kavimandan, A. Gokhale, G. Karsai, and J. Gray. Managing the Quality of Software Product Line Architectures through Reusable Model Transformations. pages 13–22, June 2011. 3.4

[80] A. Kavimandan, R. Klemm, and A. Gokhale. Automated Context-sensitive Dialog Synthesis for Enterprise Workflows using Templatized Model Transformations. In *Proceedings of the 12th International Conference on Enterprise Computing (EDOC '08)*, pages 159–168, Munchen, Germany, Sept. 2008. IEEE. 4.2.1

[81] A. Kavimandan, A. Narayanan, A. Gokhale, and G. Karsai. Evaluating the Correctness and Effectiveness of a Middleware QoS Configuration Process in Distributed Real-time and Embedded Systems. In *Proceedings of the* 11th *IEEE International Symposium on Object-oriented Real-time distributed Computing (ISORC 2008)*, pages 100–107, Orlando, FL, USA, May 2008. 4.2.1, 4.2.1

[82] S. Khare, K. An, S. Tambe, A. Gokhale, and A. Meena. Industry Paper: Reactive Stream Processing for Data-centric Publish/Subscribe. In *The 9th ACM International Conference on Distributed Event-Based Systems (DEBS' 15)*, pages 234–245, Oslo, Norway, July 2015. ACM. 2, 3.3

[83] A. Kogekar, D. Kaul, A. Gokhale, P. Vandal, U. Praphamontripong, S. Gokhale, J. Zhang, Y. Lin, and J. Gray. Model-driven Generative Techniques for Scalable Performabality Analysis of Distributed Systems. In *Proceedings of the NSF NGS Workshop, International Conference on Parallel and Distributed Processing Symposium (IPDPS) 2006*, Rhodes Island, Greece, April 2006. IEEE. 4.2.1

[84] J. Koomey. Growth in data center electricity use 2005 to 2010. *A report by Analytical Press, completed at the request of The New York Times*, 2011. 3.1

[85] A. S. Krishna, E. Turkay, A. Gokhale, and D. C. Schmidt. Model-Driven Techniques for Evaluating the QoS of Middleware Configurations for DRE Systems. In *Proceedings of the 11th Real-time Technology and Application Symposium (RTAS '05)*, pages 180–189, San Francisco, CA, Mar. 2005. IEEE. 4.2.1, 4.2.1

[86] A. S. Krishna, C. Yilmaz, A. Porter, A. Memon, D. C. Schmidt, and A. Gokhale. Distributed continuous quality assurance process for evaluating qos of performance intensive software. *Studia Infomatica Universalis*, 4, Mar. 2005. 4.2.1

[87] Y. Li, S. Shekhar, Y. Vorobeychik, X. Koutsoukos, and A. Gokhale. Simulation-based Optimization as a Service for Dynamic Data-driven Applications Systems. In *To Appear in the 1st Conference on Dynamic Data Driven Applications and Systems (DDDAS)*, Hartford, CT, USA, Aug. 2016. Springer. 2

[88] LXC. Linux Container, 2014. Last accessed: 10/11/2014. 3.1.5

[89] N. Lynch and M. Tuttle. An Introduction to Input/Output Automata. *CWI-Quarterly*, 2(3):219–246, September 1989. 4.2.1

[90] I. Maier and M. Odersky. Deprecating the Observer Pattern with Scala.react. Technical report, 2012. 3.3

[91] V. Mauch, M. Kunze, and M. Hillenbrand. High Performance Cloud Computing. *Future Generation Computer Systems*, 29(6):1408–1416, 2013. 3.1.5

[92] L. A. Meyerovich, A. Guha, J. Baskin, G. H. Cooper, M. Greenberg, A. Bromfield, and S. Krishnamurthi. Flapjax: A Programming Language for Ajax Applications. In *ACM SIGPLAN Notices*, volume 44, pages 1–20. ACM, 2009. 3.3

[93] J. Muppala, G. Ciardo, and K. S. Trivedi. "Stochastic reward nets for reliability prediction". *Communications in Reliability, Maintainability and Serviceability: An Intl. Journal Published by SAE International*, 1(2):9–20, July 1994. 4.2.1

[94] B. Natarajan, A. Gokhale, D. C. Schmidt, and S. Yajnik. Applying Patterns to Improve the Performance of Fault-Tolerant CORBA. In *Proceedings of the* 7^{th} *International Conference on High Performance Computing (HiPC 2000)*, Bangalore, India, Dec. 2000. ACM/IEEE. 5.2

[95] B. Natarajan, A. Gokhale, D. C. Schmidt, and S. Yajnik. DOORS: Towards High-performance Fault-Tolerant CORBA. In *Proceedings of the* 2^{nd} *International Symposium on Distributed Objects and Applications (DOA 2000)*, Antwerp, Belgium, Sept. 2000. OMG. 5.2

[96] Object Management Group. *Fault Tolerant CORBA Specification*, OMG Document orbos/99-12-08 edition, Dec. 1999. 5.2

[97] Object Management Group. *Real-time CORBA Specification*, OMG Document formal/05-01-04 edition, Aug. 2002. 6.2

[98] Object Management Group. *The Common Object Request Broker: Architecture and Specification Version 3.1, Part 1: CORBA Interfaces*, OMG Document formal/2008-01-04 edition, Jan. 2008. 6.2

[99] C. O'Luanaigh. Cern data center passes 100 petabytes. `http://home.web.cern.ch/about/updates/2013/02/cern-data-centre-passes-100-petabytes`, 2013. 3.1

[100] OMG. The Data Distribution Service specification, v1.2. `http://www.omg.org/spec/DDS/1.2`, 2007. 3.3

[101] W. Otte, A. Gokhale, and D. Schmidt. Efficient and Deterministic Application Deployment in Component-based, Enterprise Distributed, Real-time, and Embedded Systems. *Elsevier Journal of Information and Software Technology (IST)*, 55(2):475–488, Feb. 2013. <ce:title>Special Section: Component-Based Software Engineering (CBSE), 2011</ce:title>. 3.5

[102] W. R. Otte, A. Gokhale, and D. C. Schmidt. Predictable Deployment in Component-based Enterprise Distributed Real-time and Embedded Systems. In *Proceedings of the 14th international ACM Sigsoft symposium on Component based software engineering*, CBSE '11, pages 21–30, New York, NY, USA, 2011. ACM. 3.5

[103] W. R. Otte, A. Gokhale, D. C. Schmidt, and J. Willemsen. Infrastructure for Component-based DDS Application Development. In *Proceedings of the 10th ACM international conference on Generative programming and component engineering*, GPCE '11, pages 53–62, New York, NY, USA, 2011. ACM. 3.5

[104] P. Patil and A. Gokhale. Improving the Reliability and Availability of Vehicular Communications Using Voronoi Diagram-Based Placement of Road Side Units. In *Poster Session of the 31st IEEE International Symposium on Reliable Distributed Systems (SRDS '12)*, pages 400–401, Irvine, CA, USA, Oct. 2012. IEEE. 3.5

[105] P. Patil, A. Hakiri, Y. Barve, and A. Gokhale. Enabling Software-Defined Networking for Wireless Mesh Networks in Smart Environments. In *15th IEEE International Symposium on Network Computing and Applications (NCA 2016)*, pages 153–157, Boston, MA, USA, Oct. 2016. IEEE. 2, 3.2

[106] P. Patil, A. Hakiri, and A. Gokhale. Bootstrapping Software Defined Network for Flexible and Dynamic Control Plane Management. In *1st IEEE Conference on Network Softwarization*, pages 1–5, London, UK, Apr. 2015. IEEE. 2, 3.2

[107] P. Patil, A. Hakiri, and A. Gokhale. Adaptive Software-defined Multicast for Efficient Data Center Group Communications. In *Submitted to IEEE Conference on Communications (ICC)*, Paris, France, May 2017. IEEE. 3.2

[108] I. Pembeci, H. Nilsson, and G. Hager. Functional Reactive Robotics: An exercise in Principled Integration of Domain-specific Languages. In *Proceedings of the 4th ACM SIGPLAN international conference on Principles and practice of declarative programming*, pages 168–179. ACM, 2002. 3.3

[109] L. Poff, A. Gokhale, and M. McDonald. A Framework for Broker Placement in Vehicular Ad hoc Networks. In *Special Session on Collaboration for Dynamic Resource Management in Mobile P2P Networks (CDRM '12), International Conference on Collaboration Techniques and Systems (CTS '12)*, pages 182–189, Denver, CO, USA, May 2012. ACM, IEEE, IFIP. 3.5

[110] L. Poff, M. McDonald, and A. Gokhale. A CPS Framework for Decision-Making in Intelligent Transportation Systems. In *Poster at Distributed Event-based Systems (DEBS) 2011*, pages 395–396, Yorktown Heights, NY, USA, July 2011. ACM. 3.5

[111] S. Pradhan, A. Dubey, and A. Gokhale. Designing a Resilient Deployment and Reconguration Infrastructure for Remotely Managed Cyber-Physical Systems. In *8th International Workshop on Software Engineering for Resilient Systems (SERENE)*, pages 88–104, Gothenberg, Sweden, Sept. 2016. 2

[112] S. Pradhan, A. Dubey, A. Gokhale, and M. Lehofer. CHARIOT: A Domain Specific Language for Extensible Cyber-Physical Systems. In *Proceedings of the 15th Domain-specific Modeling Workshop, Companion Proceedings of the 2015 ACM SIGPLAN International Conference on Systems, Programming, Languages and Applications: Software for Humanity (SPLASH)*, Pittsburg, PA, USA, Oct. 2015. ACM. 2

[113] S. Pradhan, A. Dubey, A. Gokhale, and M. Lehofer. Wip Abstract: Platform for Designing and Managing Resilient and Extensible CPS. In *ACM/IEEE International Conference on Cyber Physical Systems (ICCPS)*, page 1, Vienna, Austria, Apr. 2016. ACM/IEEE. 2

[114] S. Pradhan, A. Gokhale, W. Otte, and G. Karsai. Real-time Fault-tolerant Deployment and Configuration Framework for Cyber Physical Systems. In *Work-in-Progress Session at the 33rd IEEE Real-time Systems Symposium (RTSS '12)*, San Juan, Puerto Rico, USA, Dec. 2012. IEEE. 3.5

[115] U. Praphamontripong, S. Gokhale, A. Gokhale, and J. Gray. Performance Analysis of an Asynchronous Web Server. In *Proceedings of the 30th Annual International Conference on Computer Software and Applications (COMPSAC 06)*, pages 22–25, Chicago, IL, Sep 2006. 4.2.1

[116] U. Praphamontripong, S. Gokhale, A. Gokhale, and J. Gray. Performance Analysis of a Middleware Demultiplexing Pattern. In *Proceedings of Hawaii International Conference on System Sciences HICSS-40 (2007), Minitrack, Software Technology Track*, Big Island, Hawaii, Jan 2007. 4.2.1

[117] I. Pyarali, C. O'Ryan, D. C. Schmidt, N. Wang, V. Kachroo, and A. Gokhale. Applying Optimization Patterns to the Design of Real-time ORBs. In *Proceedings of the 5th Conference on Object-Oriented Technologies and Systems*, pages 145–159, San Diego, CA, May 1999. USENIX. 6.2

[118] I. Pyarali, C. O'Ryan, D. C. Schmidt, N. Wang, V. Kachroo, and A. Gokhale. Using Principle Patterns to Optimize Real-time ORBs. *IEEE Concurrency Magazine*, 8(1), 2000. 6.1

[119] Robby, M. Dwyer, and J. Hatcliff. Bogor: An Extensible and Highly-Modular Model Checking Framework. In *Proceedings of the 4th Joint Meeting of the European Software Engineering Conference and ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE 2003)*, Helsinki, Finland, September 2003. ACM. 4.2.1

[120] N. Roy, A. Dubey, and A. Gokhale. Efficient Autoscaling in the Cloud using Predictive Models for Workload Forecasting. In *4th IEEE International Conference on Cloud Computing (Cloud 2011)*, pages 500–507, Washington, DC, July 2011. IEEE. 3.5

[121] N. Roy, A. Dubey, A. Gokhale, and L. Dowdy. A Capacity Planning Process for Performance Assurance of Component-based Distributed Systems. In *Proceedings of the 2nd ACM/SPEC International Conference on Performance Engineering (ICPE 2011)*, pages 259–270, Karlsruhe, Germany, Mar. 2011. ACM/SPEC. 3.5

[122] N. Roy, A. Gokhale, and L. Dowdy. Impediments to Analytical Modeling of Multi-Tiered Web Applications. In *Short Paper and Poster Proceedings of the 18th Annual Meeting of the IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS '10)*, pages 441–443, Miami Beach, FL, USA, Aug. 2010. IEEE. 3.5

[123] M. Satyanarayanan. Mobile Computing: The Next Decade. *SIGMOBILE Mob. Comput. Commun. Rev.*, 15(2):2–10, Aug. 2011. 2

[124] R. E. Schantz and D. C. Schmidt. Middleware for Distributed Systems: Evolving the Common Structure for Network-centric Applications. In J. Marciniak and G. Telecki, editors, *Encyclopedia of Software Engineering*. Wiley & Sons, New York, 2001. 6.1

[125] D. C. Schmidt. Model-Driven Engineering. *IEEE Computer*, 39(2):25–31, 2006. 4.1

[126] D. C. Schmidt, A. Gokhale, T. Harrison, and G. Parulkar. A High-Performance Endsystem Architecture for Real-time CORBA. *IEEE Communications Magazine*, 14(2), Feb. 1997. 6.2

[127] D. C. Schmidt, D. L. Levine, and S. Mungee. The Design and Performance of Real-time Object Request Brokers. *Computer Communications*, 21(4):294–324, Apr. 1998. 6.2

[128] D. C. Schmidt, S. Mungee, S. Flores-Gaitan, and A. Gokhale. Software Architectures for Reducing Priority Inversion and Non-determinism in Real-time Object Request Brokers. *Journal of Real-time Systems, special issue on Real-time Computing in the Age of the Web and the Internet*, 21(2), 2001. 6.2

[129] D. C. Schmidt, B. Natarajan, A. Gokhale, N. Wang, and C. Gill. TAO: A Pattern-Oriented Object Request Broker for Distributed Real-time and Embedded Systems. *IEEE Distributed Systems Online*, 3(2), Feb. 2002. 6.2

[130] D. C. Schmidt, M. Stal, H. Rohnert, and F. Buschmann. *Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects, Volume 2*. Wiley & Sons, New York, 2000. 4.2.1

[131] A. Shah, K. An, A. Gokhale, and J. White. Maximizing Service Uptime of Smartphone-based Distributed Real-time and Embedded Systems. In *14th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing (ISORC 2011)*, pages 3–10, Newport Beach, CA, USA, Mar. 2011. IEEE. 3.5

[132] S. Shekhar, F. Caglar, K. An, T. Kuroda, A. Gokhale, and S. Gokhale. A Model-driven Approach for Price/Performance Tradeoffs in Cloud-based MapReduce Application Deployment. In *2nd International Workshop on Model-Driven Engineering for High Performance and CLoud computing (MDHPCL) at MODELS 2013*, pages 37–42, Miami Beach, FL, Sept. 2013. CEUR. 3.4

[133] S. Shekhar, F. Caglar, A. Dukeman, L. Hou, A. Gokhale, J. Kinnebrew, G. Biswas, and D. Fisher. An Evaluation of a Collaborative STEM Education Framework for High and Middle School Students. In *Poster Paper at 121st ASEE Annual Conference, K-12 and Pre-Engineering Track*. ASEE, Indianapolis, IN, USA, June 2014. 2

[134] S. Shekhar, A. Chhokra, A. Bhattacharjee, G. Aupy, and A. Gokhale. INDICES: Exploiting Edge Resources for Performance-aware Cloud-hosted Services. In *1st IEEE International Conference on Fog and Edge Computing (ICFEC)*, pages 75–80, Madrid, Spain, May 2017. IEEE. 2

[135] S. Shekhar, M. Walker, H. Abdelaziz, F. Caglar, A. Gokhale, and X. Koutsoukos. A Simulation-as-a-Service Cloud Middleware. *Journal of the Annals of Telecommunications*, 74(3-4):93–108, 2016. 3.1.5

[136] D. Synodinos. Reactive Programming as an Emerging Trend. `http://www.infoq.com/news/2013/08/reactive-programming-emerging`, 2013. 3.3

[137] J. Sztipanovits and G. Karsai. Model-Integrated Computing. *IEEE Computer*, 30(4):110–112, Apr. 1997. 4.2.1

[138] S. Tambe, J. Balasubramanian, A. Gokhale, and T. Damiano. MDDPro: Model-Driven Dependability Provisioning in Enterprise Distributed Real-Time and Embedded Systems. In *Proceedings of the International Service Availability Symposium (ISAS)*, volume 4526, pages 127–144, Durham, New Hampshire, USA, 2007. Springer. 4.2.1

[139] S. Tambe, A. Dabholkar, and A. Gokhale. CQML: Aspect-oriented Modeling for Modularization and Weaving QoS Concerns in Component-based Systems. In *Proceedings of the* 16th *IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 09)*, pages 11–20, San Francisco, CA, Apr. 2009. IEEE Computer Society. 4.2.1

[140] S. Tambe, A. Daholkar, and A. Gokhale. MoPED: A Model-based Provisioning Engine for Dependability in Component-based Distributed Real-time Embedded Systems. In *18th IEEE International Conference and Workshops on Engineering of Computer-Based Systems (ECBS 2011)*, pages 44–51, Las Vegas, NV, USA, Apr. 2011. IEEE. 3.4

[141] S. Tambe and A. Gokhale. Rectifying Orphan Components using Group-Failover in Distributed Real-time and Embedded Systems. In *14th International ACM SIGSOFT Symposium on Component-based Software Engineering (CBSE)*, pages 139–148, Boulder, CO, USA, June 2011. ACM. 3.4

[142] S. Tambe and A. Gokhale. Toward Native XML Processing Using Multi-paradigm Design in C++. In *5th Annual Boost Libraries Conference (BoostCon 2011)*, Aspen, CO, USA, May 2011. Boost. 3.4

[143] P. Tarr, H. Ossher, W. Harrison, and J. Stanley M. Sutton. N Degrees of Separation: Multi-Dimensional Separation of Concerns. In *ICSE '99: Proceedings of the International Conference on Software Engineering*, pages 107–119, May 1999. 4.2.1

[144] A. Trewyn, A. Gokhale, S. Sastry, and M. Branicky. WiP Abstract: TCP Congestion Control for Highly Available Conveyor Systems. In *Work-in-Progress Session of ACM/IEEE 3rd International Conference on Cyber-Physical Systems (ICCPS '12)*, page 212, Beijing, China, Apr. 2012. ACM/IEEE. 3.5

[145] US Environmental Protection Agency. Report to Congress on Server and Data Center Energy Efficiency: Public Law 109-431, 2007. 3.1

[146] Z. Wan, W. Taha, and P. Hudak. Event-driven FRP. In *Practical Aspects of Declarative Languages*, pages 155–172. Springer, 2002. 3.3

[147] N. Wang, D. C. Schmidt, A. Gokhale, C. Rodrigues, B. Natarajan, J. P. Loyall, R. E. Schantz, and C. D. Gill. QoS-enabled Middleware. In Q. Mahmoud, editor, *Middleware for Communications*, pages 131–162. Wiley and Sons, New York, 2004. 4.1

[148] F. Wolf, J. Balasubramanian, A. Gokhale, and D. C. Schmidt. Component Replication based on Failover Units. In *Proceedings of the 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA '09)*, pages 99–108, Aug. 2009. 4.2.2

[149] D. Wu and A. Gokhale. A Self-Tuning System based on Application Profiling and Performance Analysis for Optimizing Hadoop MapReduce Cluster Configuration. In *20th Annual IEEE International Conference on High Performance Computing (HiPC '13)*, pages 89–98, Bengaluru, India, Dec. 2013. IEEE. 3.4

[150] C. Yilmaz, A. Porter, A. S. Krishna, A. Memon, D. C. Schmidt, and A. Gokhale. Reliable Effects Screening: A Distributed Continuous Quality Assurance Process for Monitoring Performance Degradation in Evolving Software Systems. *IEEE Transactions on Software Engineering*, 33(2):124–141, Feb. 2007. 4.2.1