Performance Management Group

fortiss

München, 2015-11-05

Full-Stack Performance Model Evaluation using Probabilistic Garbage Collection Simulation

Symposium on Software Performance 2015

Felix Willnecker[1], Andreas Brunnert[1], Bernhard Koch-Kemper[2], Helmut Krcmar[2]

[1]fortiss GmbH, [2]Technical University of Munich (TUM)

fortiss GmbH
An-Institut Technische Universität München

Agenda

- Introduction
- Memory management simulation
- SPECjEnterpriseNEXT
- Evaluation
- Conclusion and Outlook

Introduction

- Performance model generators available in academia and industry[1] (Brunnert/Krcmar 2015)
- Focus on Central Processing Units (CPU) and network demands (Brunnert/Krcmar 2015, Brosig et al. 2009)

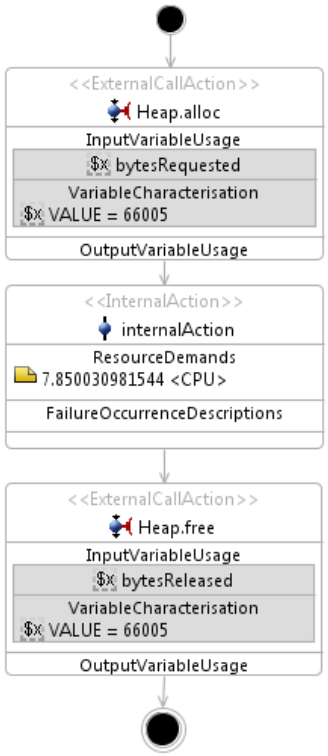
- Hard disk drive concepts exists for generators, but measurement approach and evaluation missing
- Currently available memory model focuses on static memory allocation
- Holistic performance models must cover all relevant resources of an application (Speitkamp/Bichler 2010)

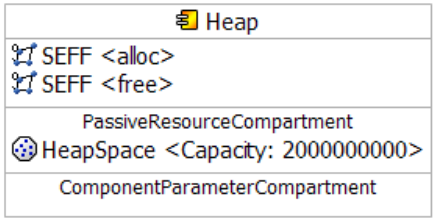
- Heap and garbage collection model extension missing
- Closing the gap with extensions to the RETIT performance model generator and full-stack evaluation based on SPECjEnterpriseNEXT

[1]https://www.retit.de/

Memory Management Model

Using PassiveResources in PCM

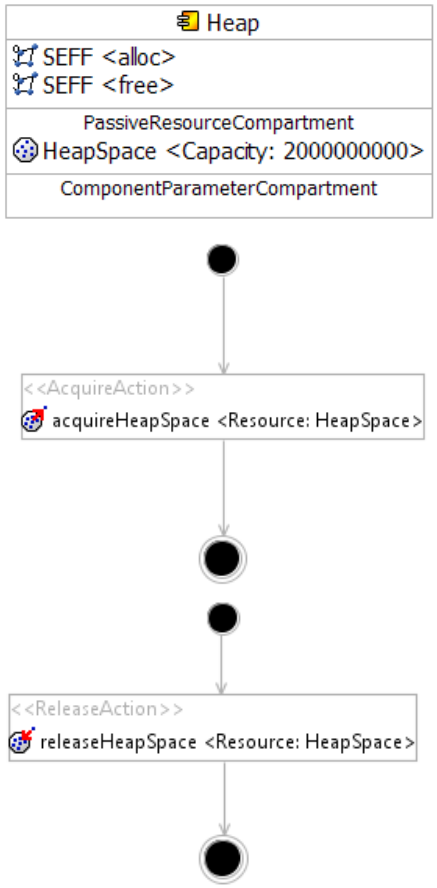




- Each *CompositeStructure* contains a Heap component
- This Heap component manages the total memory capacity as *PassivResource*
- Each *SEFF* begins with an *ExternalCallAction* allocating the mean amount of Heap allocated
- Each *SEFF* ends with *ExternalCallAction* releasing the same amount of Heap again

Static Memory Allocation

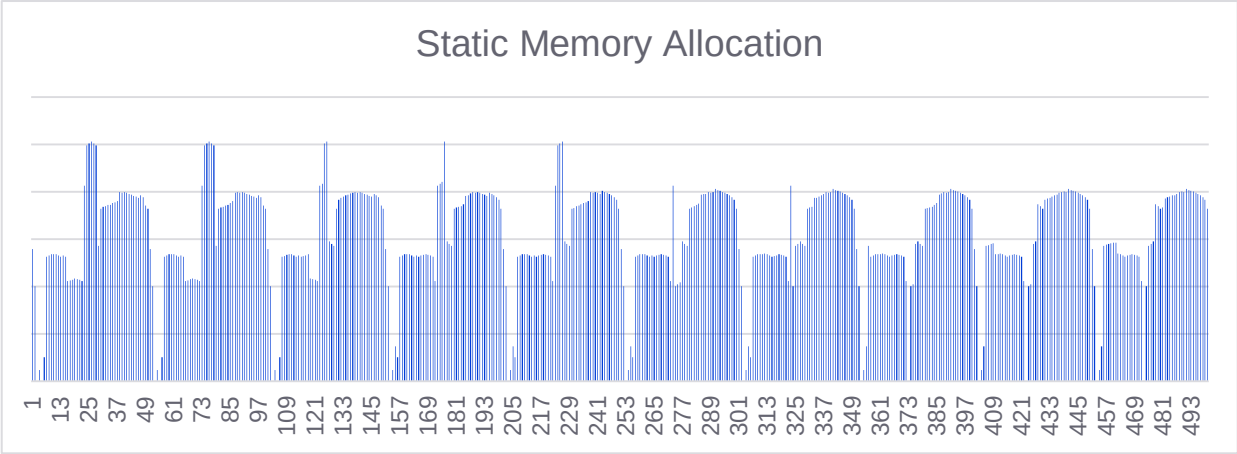
Using PassiveResources in PCM

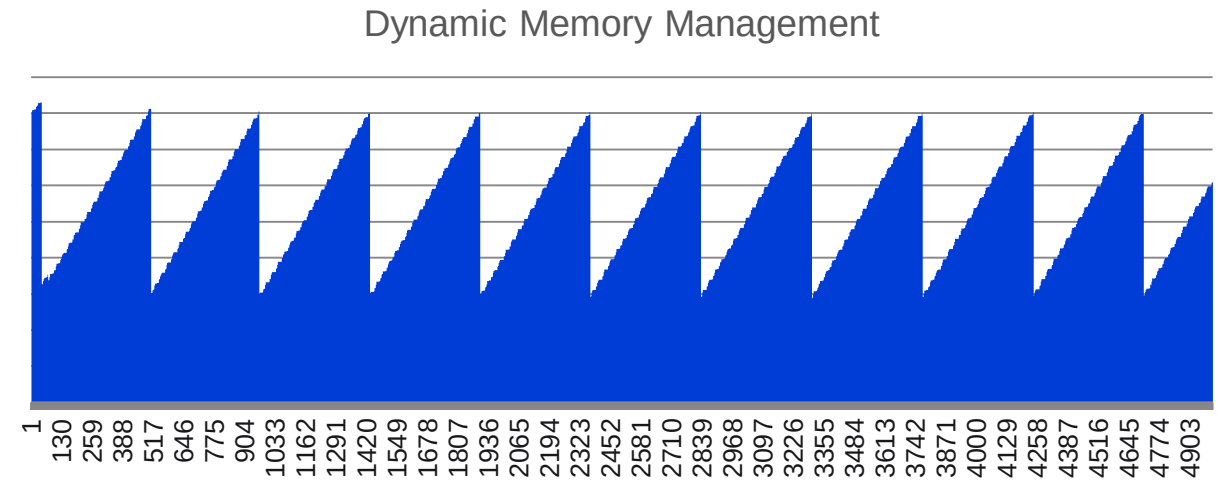


- Allocation and release is modeled using *AcquireActions* and *ReleaseActions*
- Both Actions are extended using *ParemetricResourceDemands*
- The demand contains the number of bytes allocated or released

Simulating Static and Dynamic Memory Models

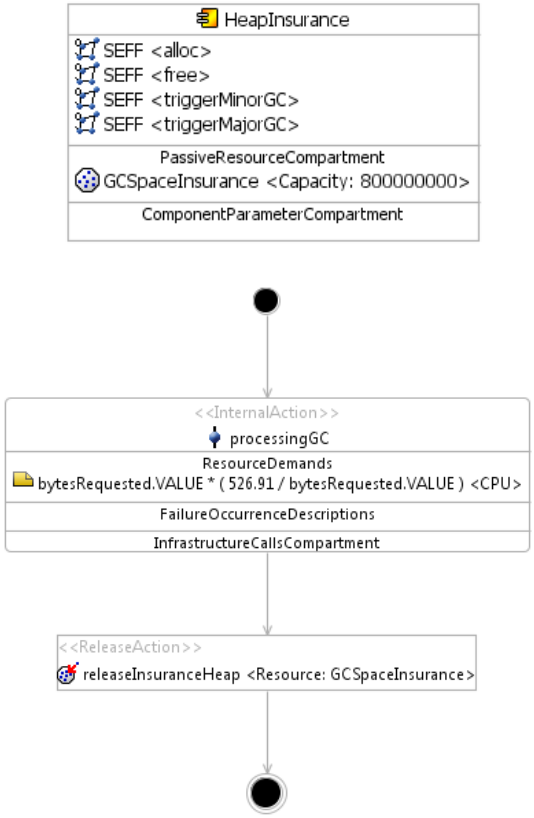
Using PassiveResources in PCM





Dynamic Memory Management

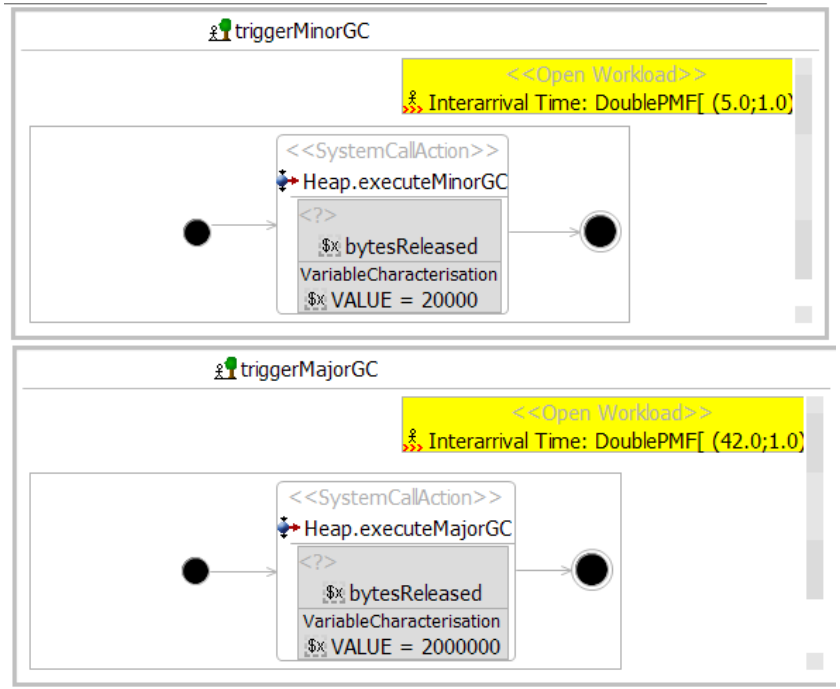
Using PassiveResources in PCM



- Added two new operations
  - *triggerMinorGC* for minor garbage collection events
  - *triggerMajorGC* for major garbage collection events
- Using *GarbageCollectionListener* for monitoring garbage collection events in JavaEE
- CPU usage depending on the number of bytes released

Dynamic Memory Management

Trigger from usage model



- Measured mean time between major and minor garbage collection events
- Trigger release from usage scenarios using an *OpenWorkload*
- Parameter of the call is the mean bytes released per major or minor garbage collection event

SPECjEnterpriseNEXT

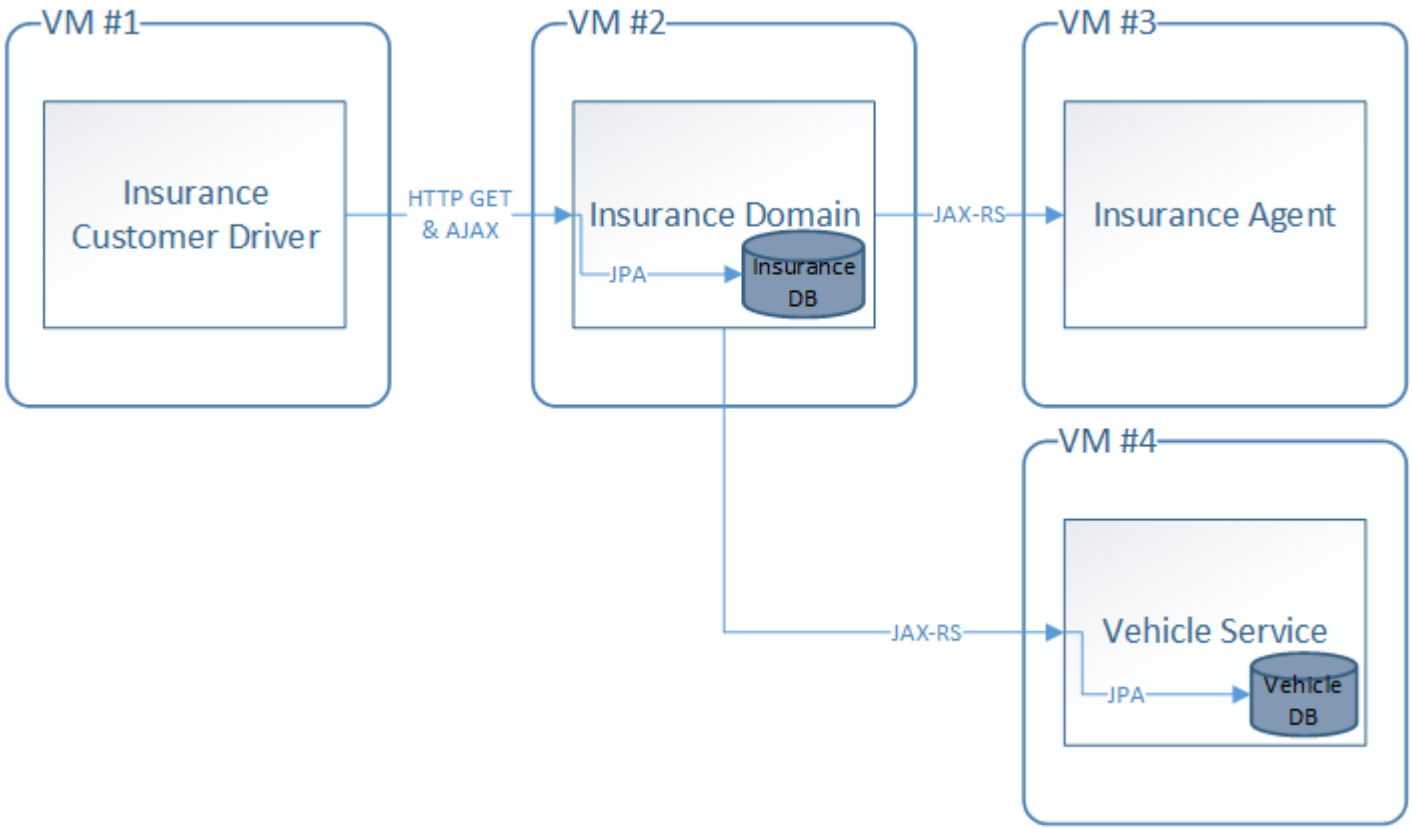
- Pre-Release version of SPECjEnterpriseNEXT used for evaluation
- Insurances for vehicles are calculated by this benchmark

- A vehicle service holds available vehicles
- An agent service calculates quotes based on customer and vehicle
- HTML/AJAX based user interface allows registration, adding vehicles and calculating quotes by calling the two web services
- Purchase not yet implemented

- Installation on three VMs using Wildfly 8.1
- Faban 1.3 based Insurance driver part of the benchmark
- Evaluation conducted with 100-160 concurrent users

SPECjEnterpriseNEXT

Evaluation deployment



Evaluation

CPU Utilization Replay

Resource	User	Metric	Insurance Domain	Vehicle Service	Insurance Agents
CPU	100	Measured utilization	43.80%	51.57%	46.13%
		Simulated utilization	40.48%	48.29%	43.04%
		Relative error	**7.58%**	**6.38%**	**6.70%**
	160	Measured utilization	70.19%	77.91%	71.73%
		Simulated utilization	64.75%	77.25%	68.88%
		Relative error	**7.75%**	**0.85%**	**3.97%**
Heap	100	Measured utilization	1458.58 MB	1435.41 MB	-
		Simulated utilization	1299.22 MB	1319.41 MB	-
		Relative error	**10.93%**	**8.08%**	-
	160	Measured utilization	1360.19 MB	1304.46 MB	-
		Simulated utilization	1296.29 MB	1317.88 MB	-
		Relative error	**4.70%**	**1.03%**	-
HDD	100	Measured utilization	0.23%	0%	0%
		Simulated utilization	0.21%	0%	0%
		Relative error	**10.72%**	0%	0%
	160	Measured utilization	0.34%	0%	0%
		Simulated utilization	0.35%	0%	0%
		Relative error	**0.65%**	0%	0%

Conclusion & Outlook

- Holistic performance model containing
  - CPU, Network, Heap (incl. GC) and HDD used
  - Deployment Units considered in distributed setup
- Probabilistic approach delivers sufficient results for Heap simulation
- Outlook
  - *ActiveResource* instead of *PassiveResource* in PCM => Reducing the number of measurement probes
  - Garbage collection space simulation instead of mean bytes released
  - Garbage collection threshold instead of interarrival time
  - Evaluation with different use cases

References

- **Brunnert, A.; Krcmar, H. (2015)**: Continuous Performance Evaluation and Capacity Planning Using Resource Profiles for Enterprise Applications. In: Journal of Systems and Software, (2015).
- **Brosig, F.; Kounev, S.; Krogmann, K. (2009)**: Automated Extraction of Palladio Component Models from Running Enterprise Java Applications. Paper presented at the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools, pp. 10:1-10:10.
- **Speitkamp, B.; Bichler, M. (2010)**: A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers. In: Services Computing, IEEE Transactions on, Vol. 3 (2010) No. 4, pp. 266-278.

Felix Willnecker

willnecker@fortiss.org

performancegroup@fortiss.org

pmw.fortiss.org