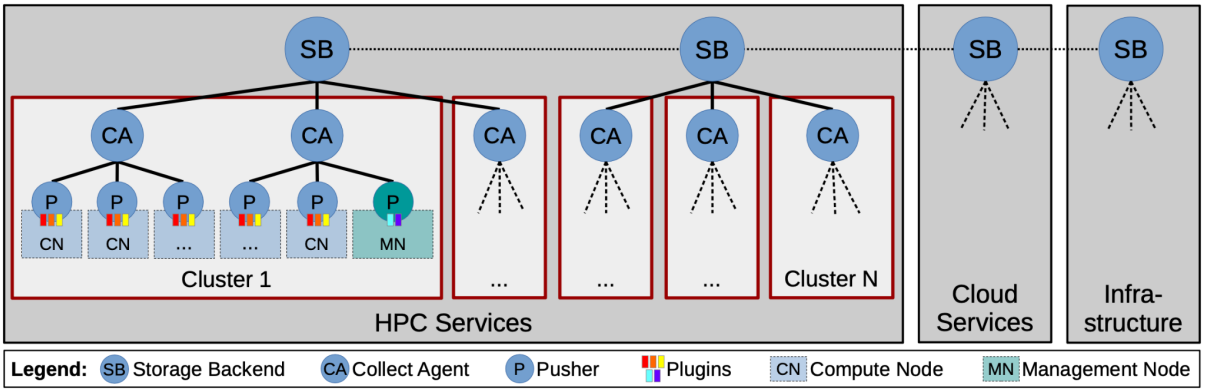
Steven Pan, 20. Januar 2022

DCDB und EAR

Data Center Data Base (DCDB)

- Open Source Monitoringlösung des LRZ, 2019
- Integration von HPC mit Data Center und Resource Management Systemen
- Sammeln, Verwalten und Speichern von Gebäude-, System- und Applikationsdaten



Quelle: Netti et. al, From Facility to Application Sensor Data

- **Pusher** sammeln und veröffentlichen **Plugin** Daten (*Sensors*) mit MQTT
- **Collector Agents** sammeln Sensors von Pushern und verteilen diese an Subscriber
- MQTT als Transportprotokoll, Publish-Subscribe Modell mit **Hierarchie**
- **Storage Backend** als Subscriber aller Pusher-Topics, Apache Cassandra: verteilte DB
- Evaluiert auf Produktionssystemen
  - MPI-Benchmarks: Overhead im Regelfall unter 3%
  - Single Node HPL: Cores bzw. Sensors vs. Single-Core-Performance Tradeoff
  - Collector Agent CPU-Load auf separaten Nodes: 9 Cores bei $5*10^5$ Sensors/s

Examon, EEESlab 2017	**LDMS**, NCSA 2014	**Ganglia**, UCL 2004

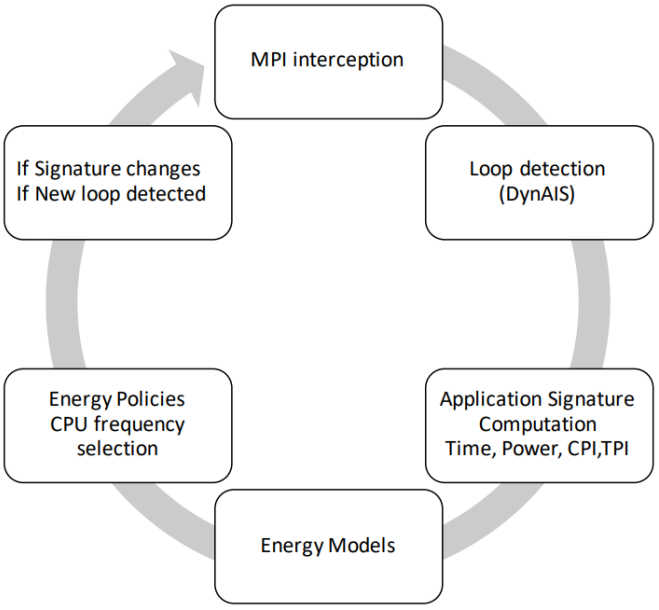
• MQTT-Broker, Hierarchie
• Cassandra DB
• Grafana
• "On-line live ML", Analysen, Power-Modeling
• Fehlende Modularität / Plugins (Nach DCDB Autoren)
• CINECA Eurora und GALILEO
• Collector auf: 8 Nodes, 128 Cores
  ○ 0.02 Hz: < 10% core
  ○ 1 Hz: < 1% core
	• Plugin basiert
	• Hierarchische Aggregation
	• Pull - RDMA, IB, Socket
	• Storage Plugins: MySQL, flat file, proprietäres Format
	• Kein Fokus auf individuelle Anpassungen (DCDB Autoren)
	• Blue Waters: 27.648 Nodes, 392.032 Cores
	• Einfluss auf Blue Waters, SNL Capacity Systems: "nicht signifikant" bis 1Hz
		• Vorgegebene und applikations-basierte Metriken
		• Multicast zwischen Nodes
		• Tree-based zwischen Clustern
		• Daten in XDR form
		• RRDtool: Visualisierung, Speicherung
		• "Läuft auf tausenden Clustern"
		• Overhead evaluiert bis 2000 Nodes und 42 Cluster
		  ○ Lokal: < 0.4%
		  ○ Aggregation: < 1.1%

Energy Aware Runtime (EAR)

- Energy Management Framework von BSC + Lenovo, 2017
- Energy-Kontrolle, -Monitoring, -**Optimierung** (MPI+OpenMP)
- Läuft auf SURF Innovation Labs Snellius und seit 2019 auf LRZ SuperMUC(-NG)

Steven Pan, 20. Januar 2022

EAR Library (EARL)



Quelle: Corbalan et. al, Energy Optimization and Analysis with EAR

1. Dynamic Application Iterative Structure Detection Algorithm (**DynAIS**)
    - Vergleicht EventID **für jeden MPI-Call** mit *N* letzten Calls
    - Gibt Status NEW_LOOP, NEW_ITERATION, END_LOOP zurück
2. Erhebung der **Applikations-Signatur** (CPI, TPI, Power, Time) pro Loop-Phase
3. Anwendung des Energie- bzw. **Zeit-Modells** (**System-Signatur** A_i...F_i für Frequenzen f_i bei Installation mit Benchmarks und linearer Regression erlernt)
4. Evaluierung der gewählten **Energy-Policy** und **Wahl der Frequenz**
    - MIN_ENERGY: *perf_degr_threshold* >= (*TimeNew* - *Time*) / *Time*
    - MIN_TIME: *min_ratio_threshold* <= *PerfGain* / *FreqGain*
5. **Validierung der Frequenzwahl**, Energie- und Zeitprojektion in nächster Loop-Phase

- Energie-Modell stammt von IBM LoadLeveler (2011)
    - Einführung 2019 auf LRZ SuperMUC mit 6% Stromeinsparung (200.000 €/a)
- EAR Evaluation
    - Memory-Bound: bis zu 14% Stromeinsparungen mit 1% längerer Laufzeit
    - Compute-Bound: Hohe Frequenzabhängigkeit → Kleinere Stromeinsparungen (2%)
    - Ausnahme hoher AVX512-Anteil: Vektorisierte Anweisungen weniger von niedrigeren Frequenzen betroffen, bis zu 10% Einsparung bei 3% längerer Laufzeit