sys-sage

A Unified Representation of Dynamic Topologies & Attributes on HPC Systems

Stepan Vanecek

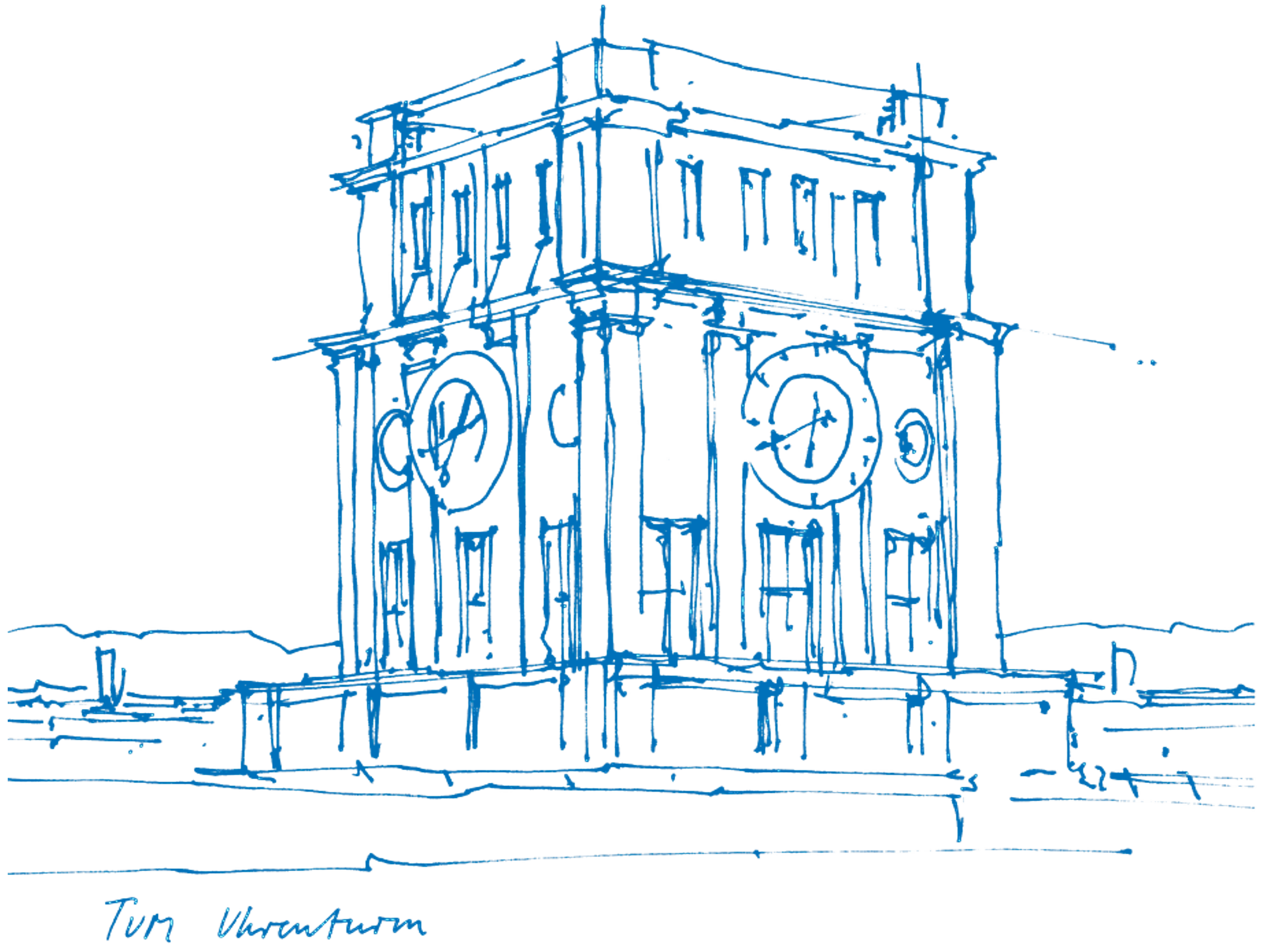
stepan.vanecek@tum.de

Martin Schulz

schulzm@in.tum.de

Chair of Computer Architecture and Parallel Systems

Technical University of Munich



ICS’24, 7th June 2024

sys-sage

A software library for managing and representing HPC system topology information

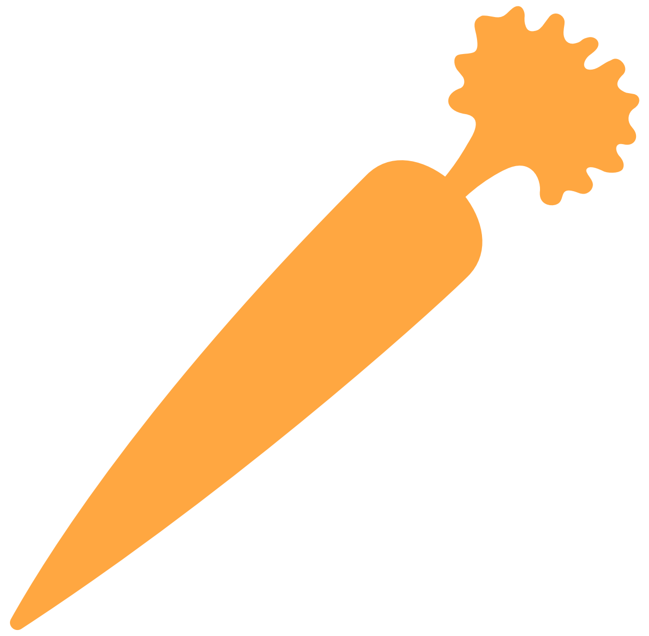
C++-based API

Main objectives

Storage and provision of all relevant information regarding

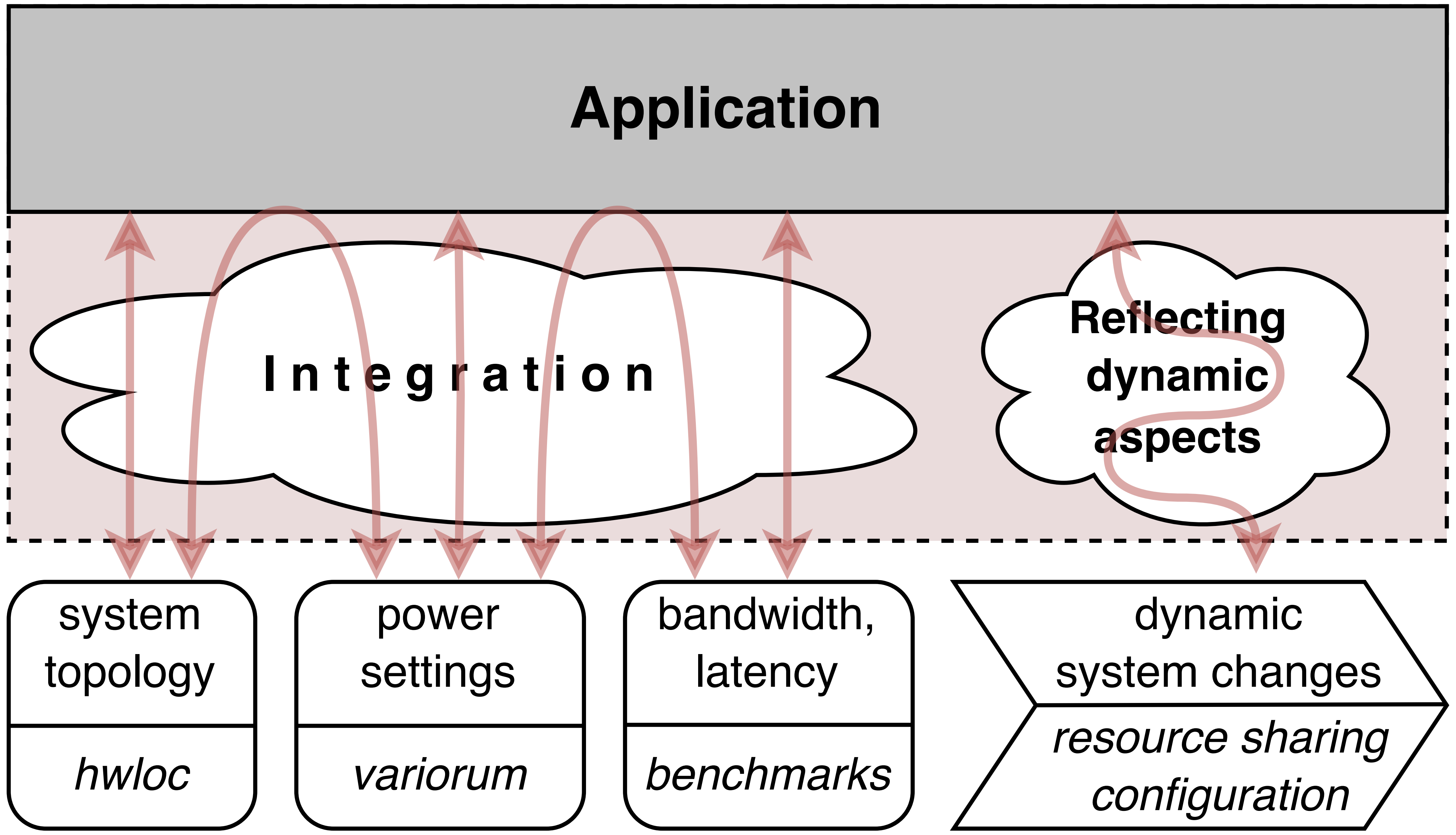
- Hardware topology,
- System configuration (both static and dynamic configuration),
- System capabilities, and
- Other data related to the HW

from different sources, logically connected to each other.

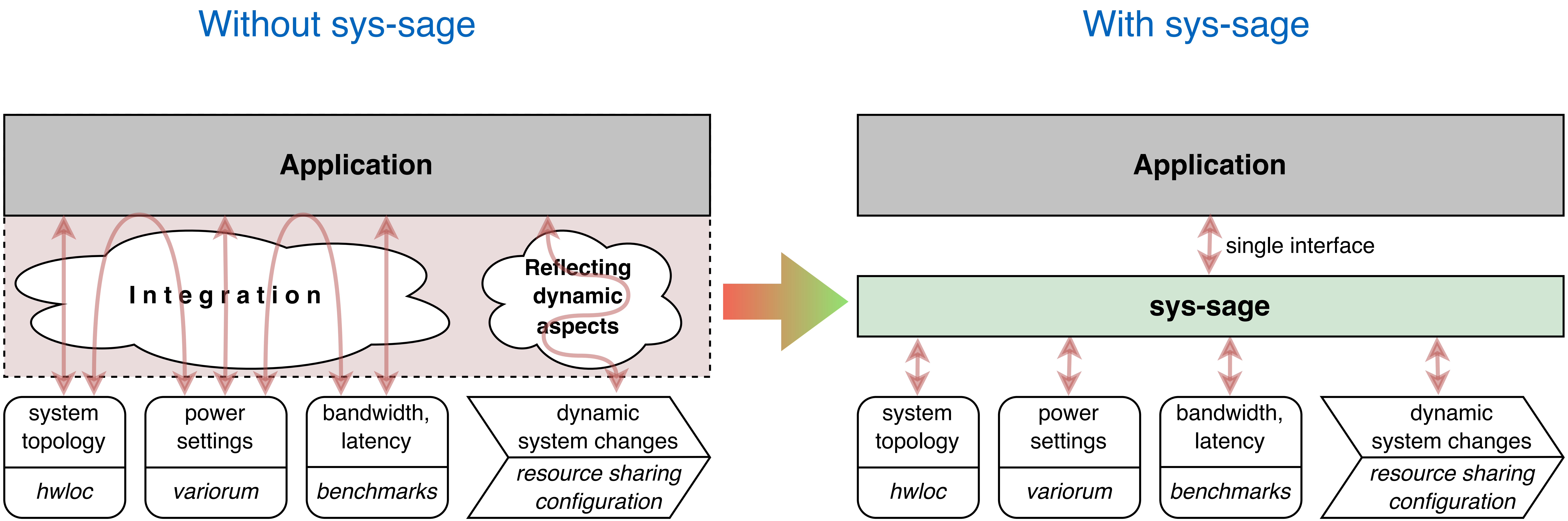


The role of sys-sage

Without sys-sage

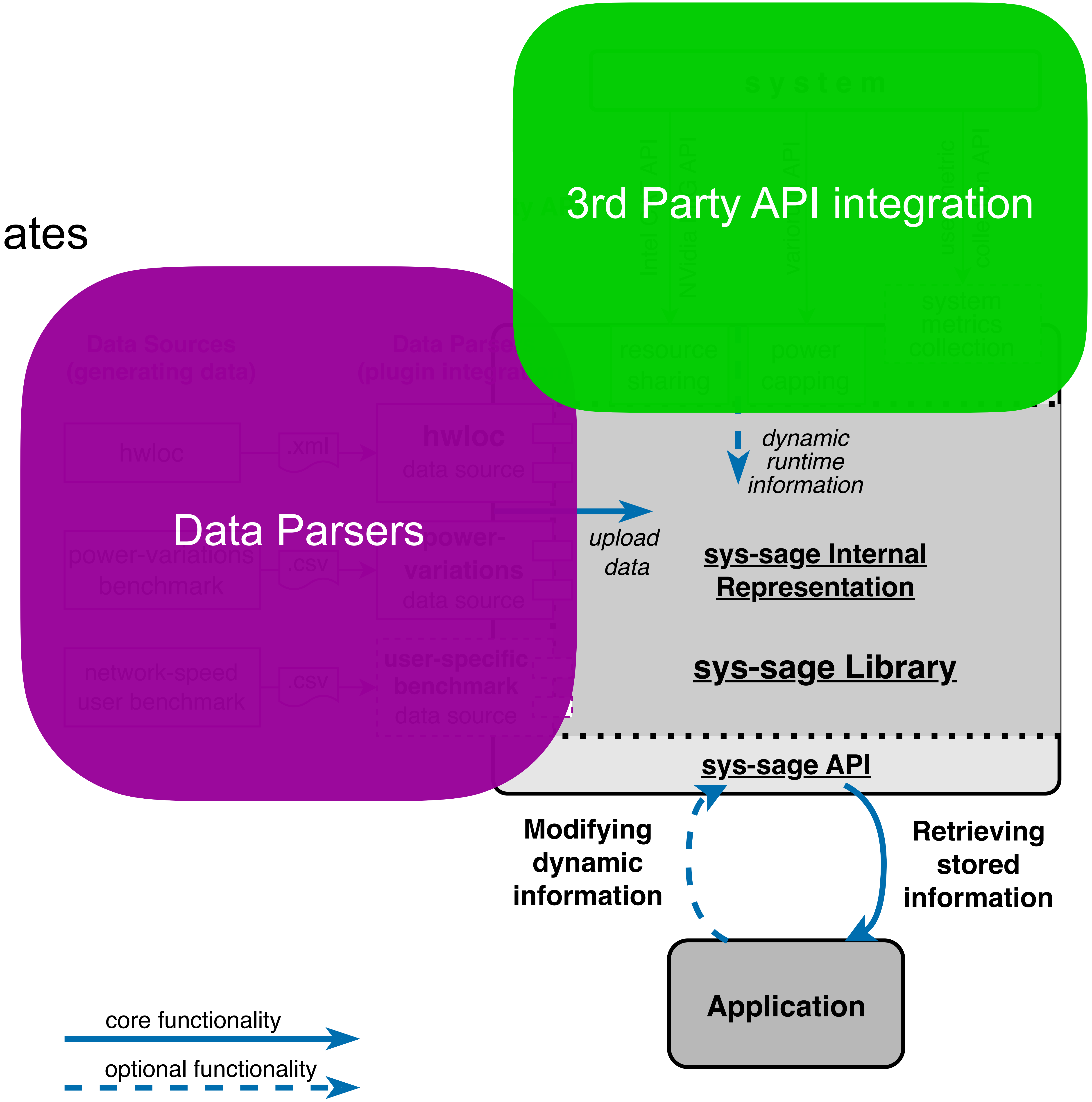


The role of sys-sage



Architecture of sys-sage

The Application triggers all data additions/updates



3rd Party API integration
system
Intel CPU API
NVidia GPU API
various API
user metric collection API
system metrics collection
resource sharing
power capping
Data Parsers
Data Sources (generating data)
Data Parsers (plugin integration)
hwloc
.xml
hwloc data source
power-variations benchmark
.csv
power-variations data source
network-speed user benchmark
.csv
user-specific benchmark data source
dynamic runtime information
upload data
sys-sage Internal Representation
sys-sage Library
sys-sage API
Modifying dynamic information
Retrieving stored information
Application
core functionality
optional functionality

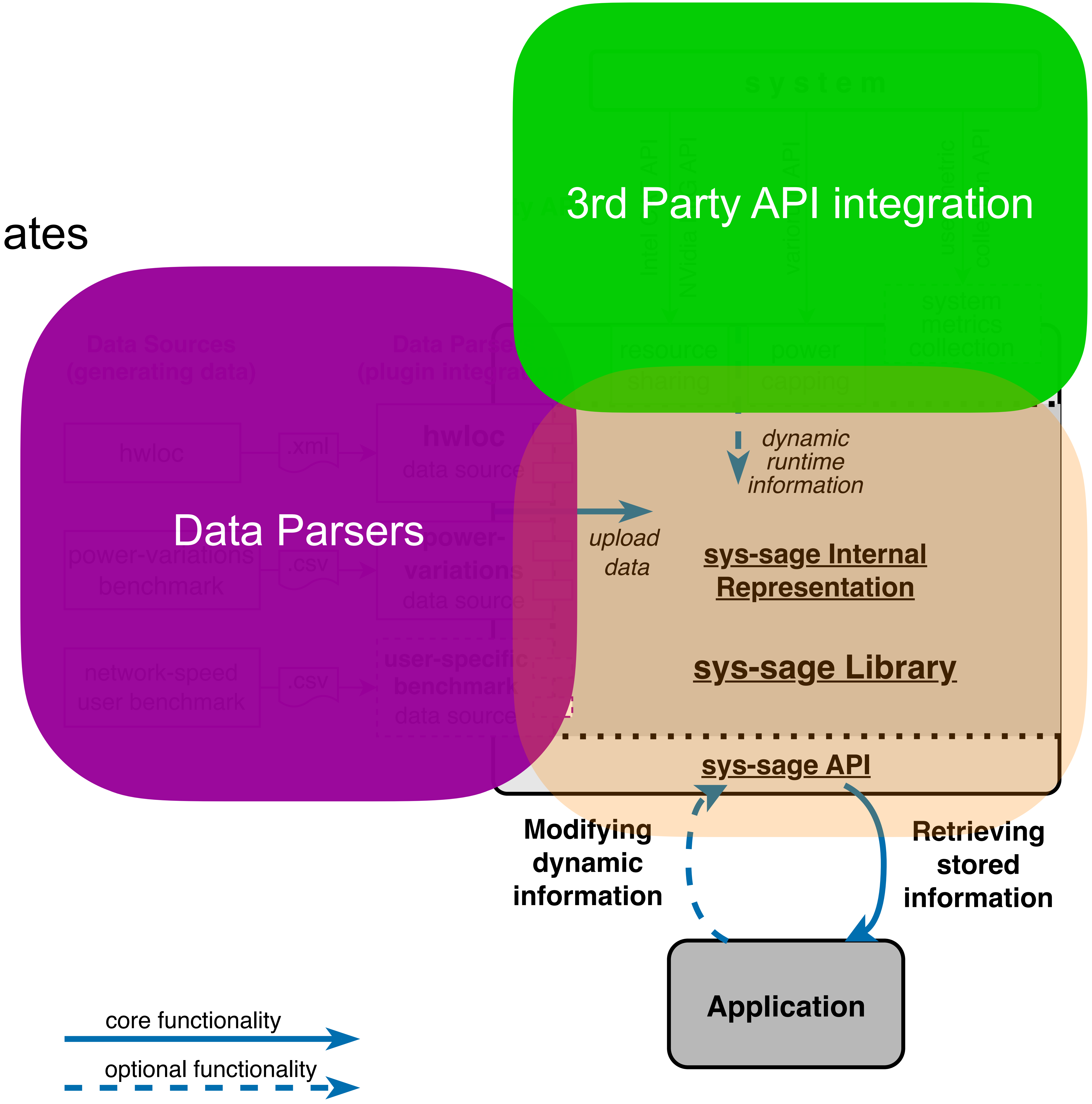
Architecture of sys-sage

The Application triggers all data additions/updates

Internal Representation

Stores all the information

- System topologies
- Logical connection of all the data
- Additional attributes


3rd Party API integration
Data Parsers
Data Sources (generating data)
Data Parsers (plugin integration)
hwloc
.xml
hwloc data source
power-variations benchmark
.csv
power-variations data source
network-speed user benchmark
.csv
user-specific benchmark data source
system
resource
power capping
system metrics collection
dynamic runtime information
upload data
sys-sage Internal Representation
sys-sage Library
sys-sage API
Modifying dynamic information
Retrieving stored information
Application
core functionality
optional functionality

sys-sage System Data Representation

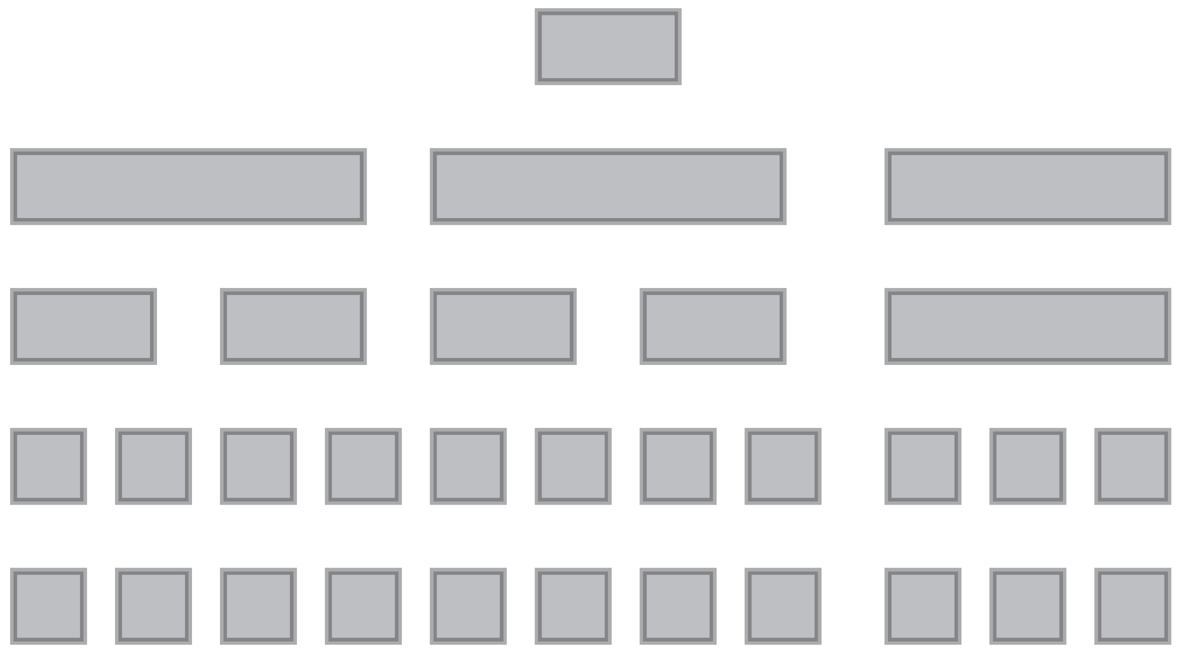
Two orthogonal concepts

- **Component Tree**
- **Data-Path Graph**

sys-sage System Data Representation

Component Tree

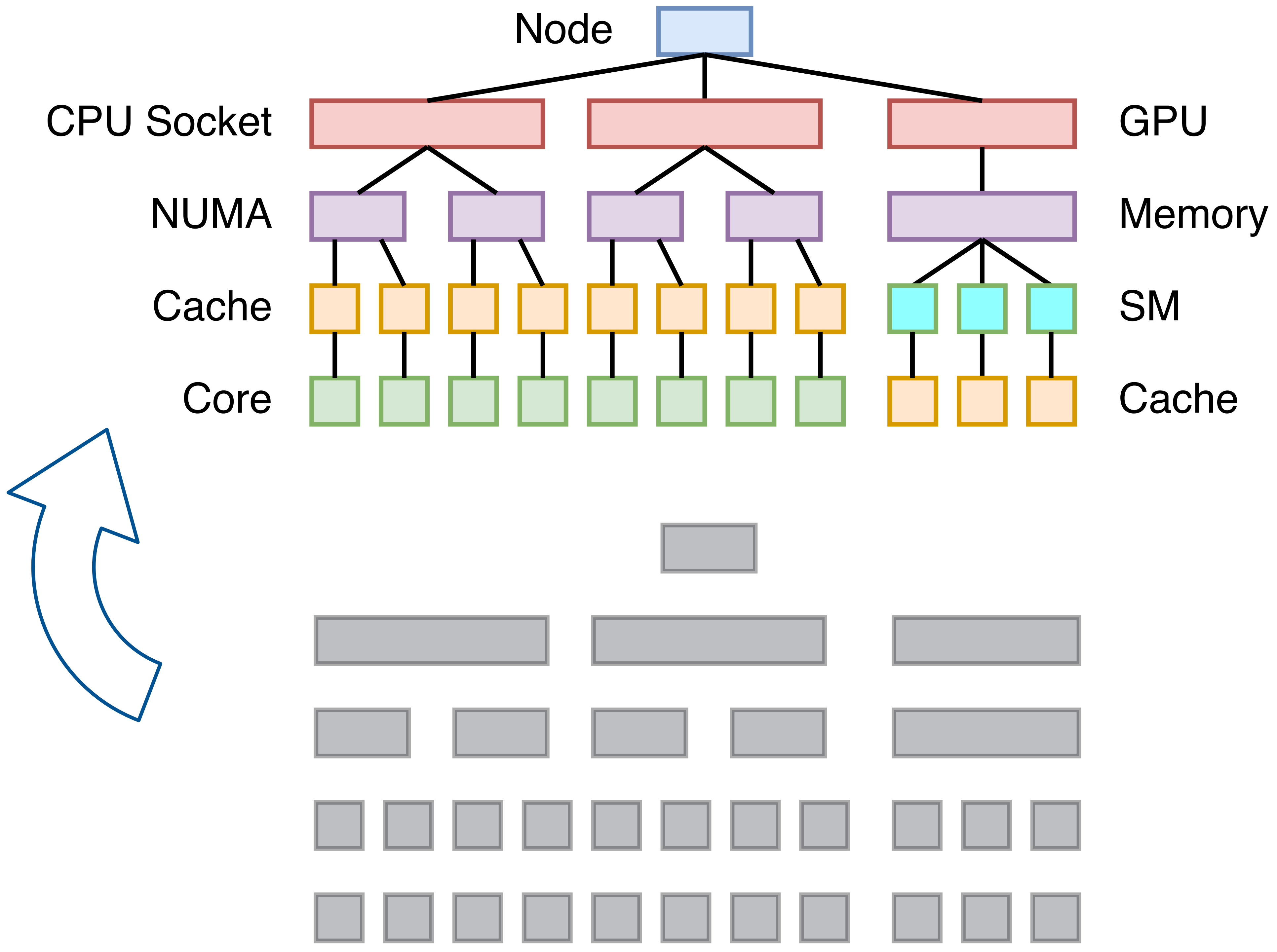
- Composed of **Components**
- Hierarchical representation (hwloc-like)
- Easy orientation
- No restrictions on the hierarchy
- *Components* contain rather static information (id, size, attributes)



sys-sage System Data Representation

Component Tree

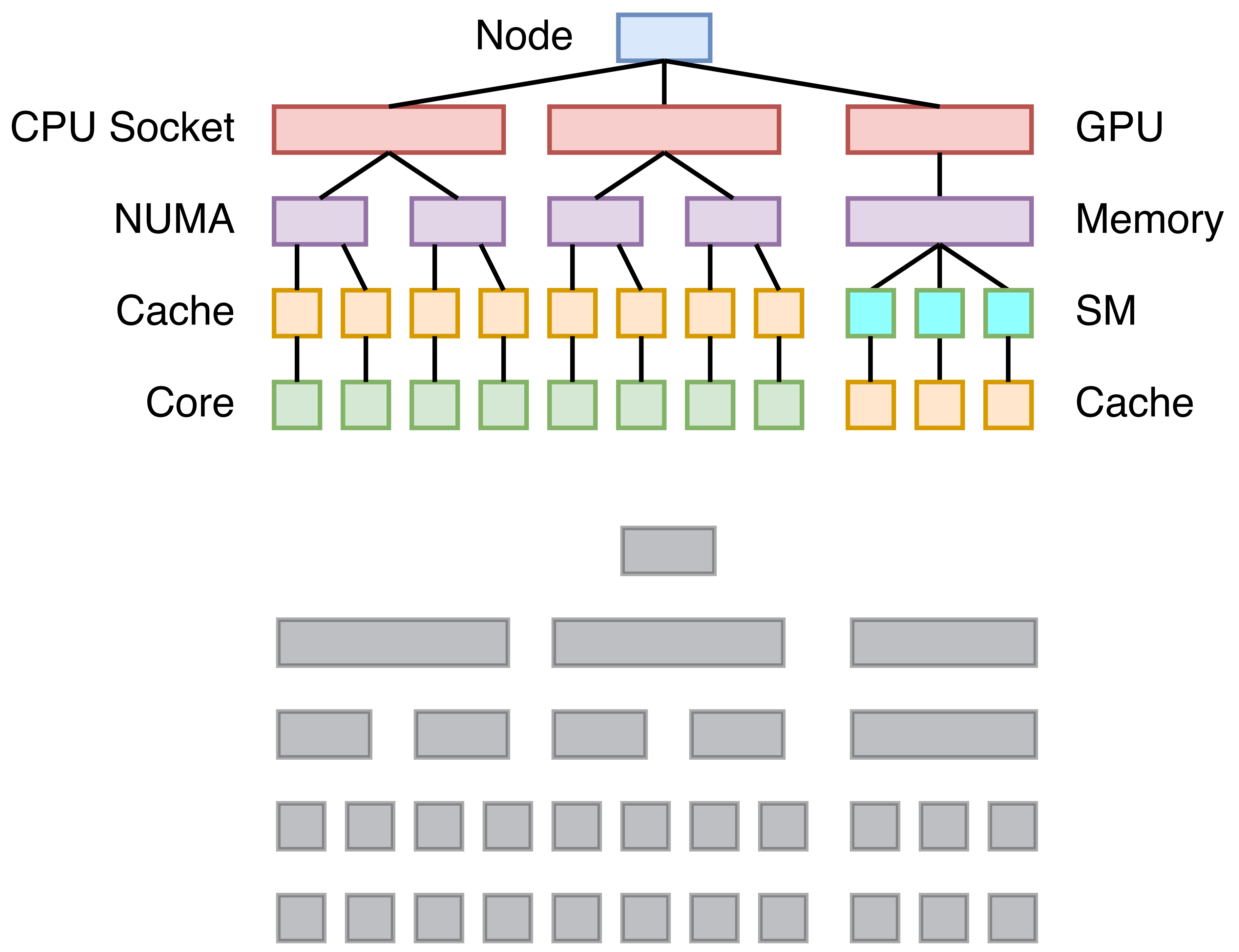
- Composed of **Components**
- Hierarchical representation (hwloc-like)
- Easy orientation
- No restrictions on the hierarchy
- *Components* contain rather static information (id, size, attributes)



sys-sage System Data Representation

Data-Path Graph

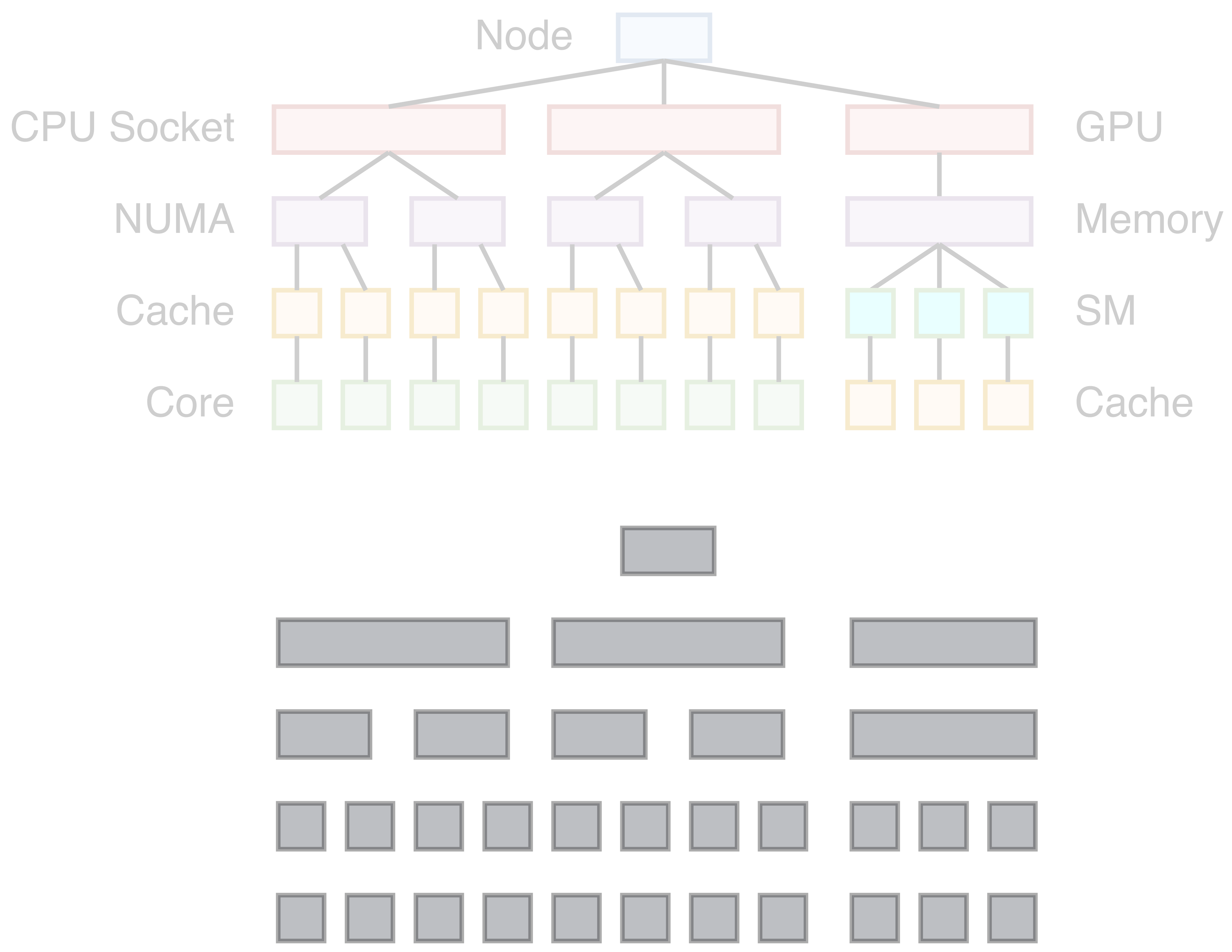
- A **Data Path** Connects two arbitrary *Components*
- Utilizes the *Component Tree* elements
- Primarily dynamic information
- *Data Paths* may contain arbitrary information
  - bandwidth, latency
  - resource partitioning configuration
  - performance/power information
  - arbitrary data, metrics, counters
  - …



sys-sage System Data Representation

Data-Path Graph

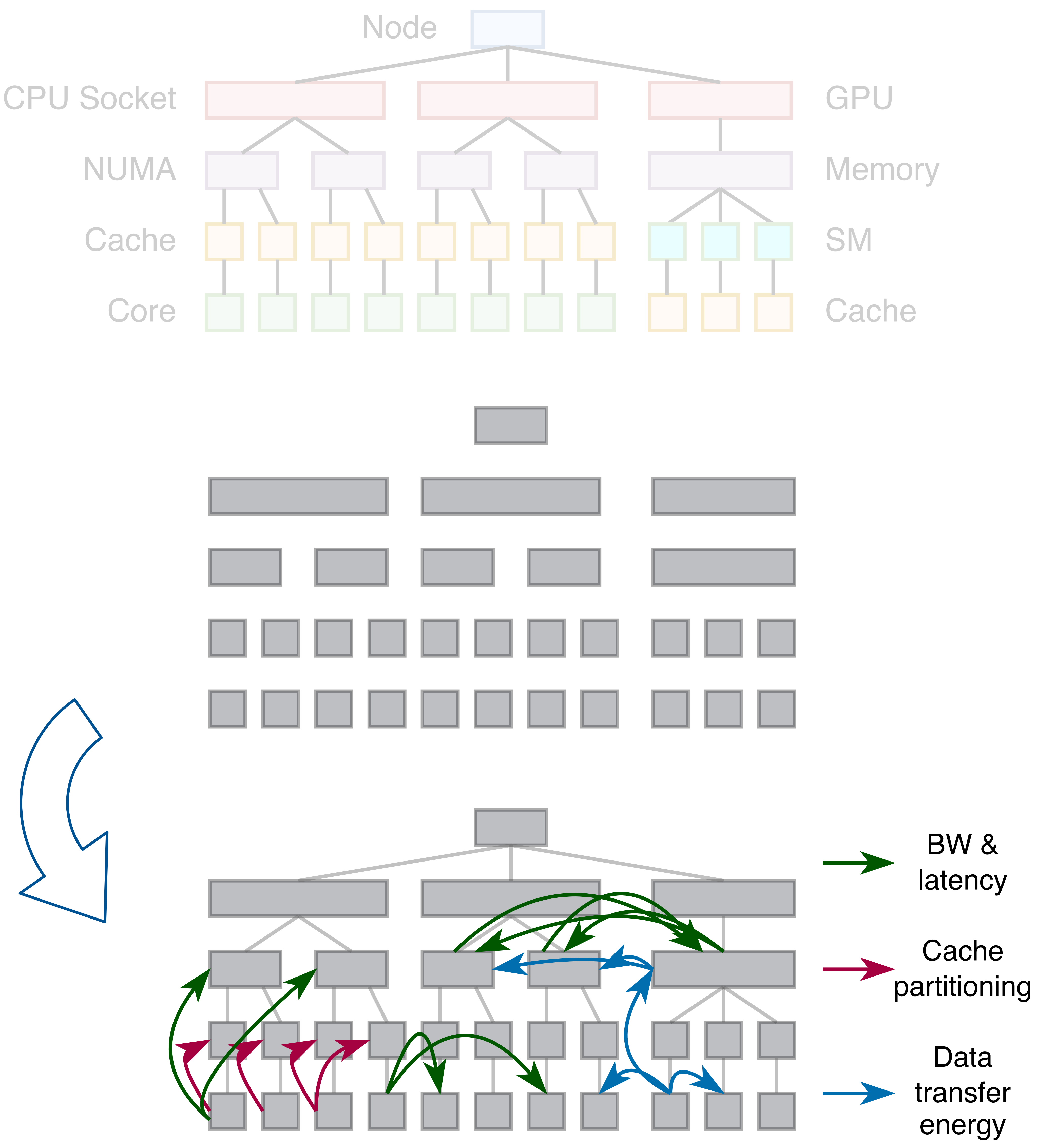
- A **Data Path** Connects two arbitrary *Components*
- Utilizes the *Component Tree* elements
- Primarily dynamic information
- *Data Paths* may contain arbitrary information
  - bandwidth, latency
  - resource partitioning configuration
  - performance/power information
  - arbitrary data, metrics, counters
  - …



sys-sage System Data Representation

Data-Path Graph

- A **Data Path** Connects two arbitrary *Components*
- Utilizes the *Component Tree* elements
- Primarily dynamic information
- *Data Paths* may contain arbitrary information
  - bandwidth, latency
  - resource partitioning configuration
  - performance/power information
  - arbitrary data, metrics, counters
  - …



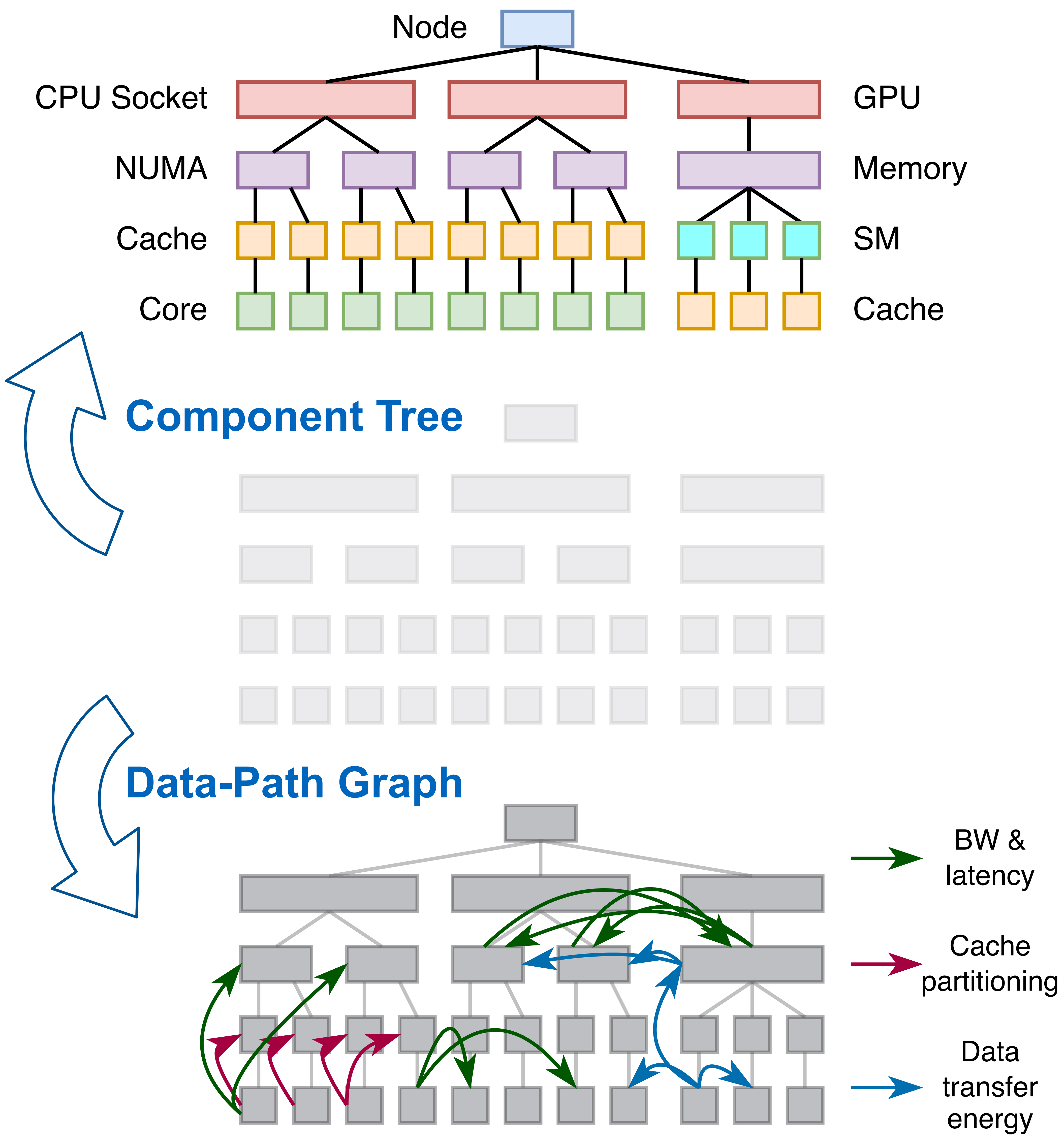
sys-sage System Data Representation

Component Tree

- Hierarchical representation (hwloc-like)
- Mandatory
- Rather static information (id, size, attributes)

Data-Path Graph

- Relation of two *Components*
- Orthogonal to the *Component Tree*
- Primarily dynamic information (BW, latency, dynamic configuration, metrics, …)



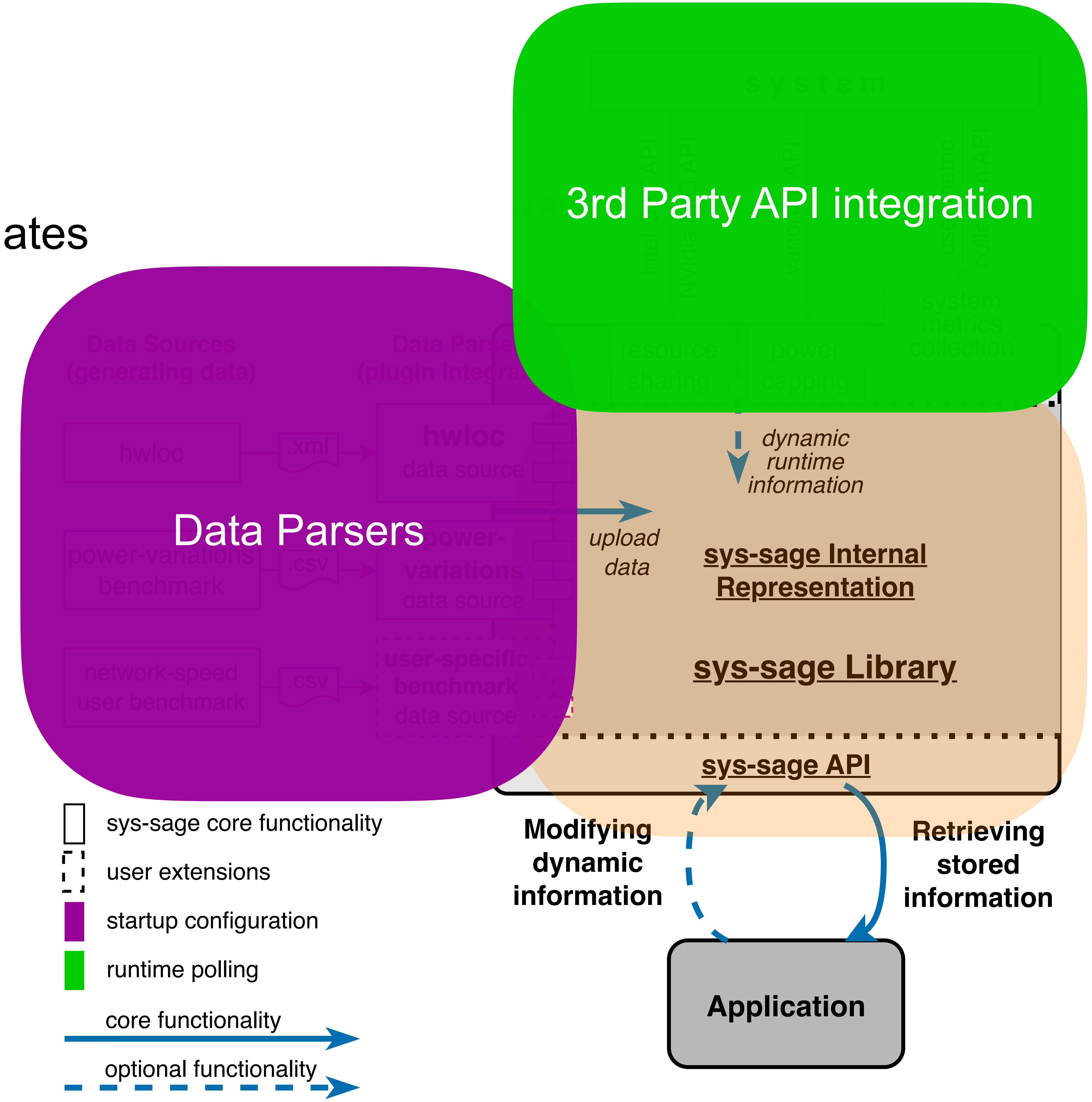
Architecture of sys-sage

The Application triggers all data additions/updates

Internal Representation

Stores all the information

- System topologies
- Logical connection of all the data
- Additional attributes


3rd Party API integration
Data Parsers
dynamic runtime information
upload data
sys-sage Internal Representation
sys-sage Library
sys-sage API
Modifying dynamic information
Retrieving stored information
Application
□ sys-sage core functionality
⬚ user extensions
🟪 startup configuration
🟩 runtime polling
core functionality
optional functionality

Architecture of sys-sage

The Application triggers all data additions/updates

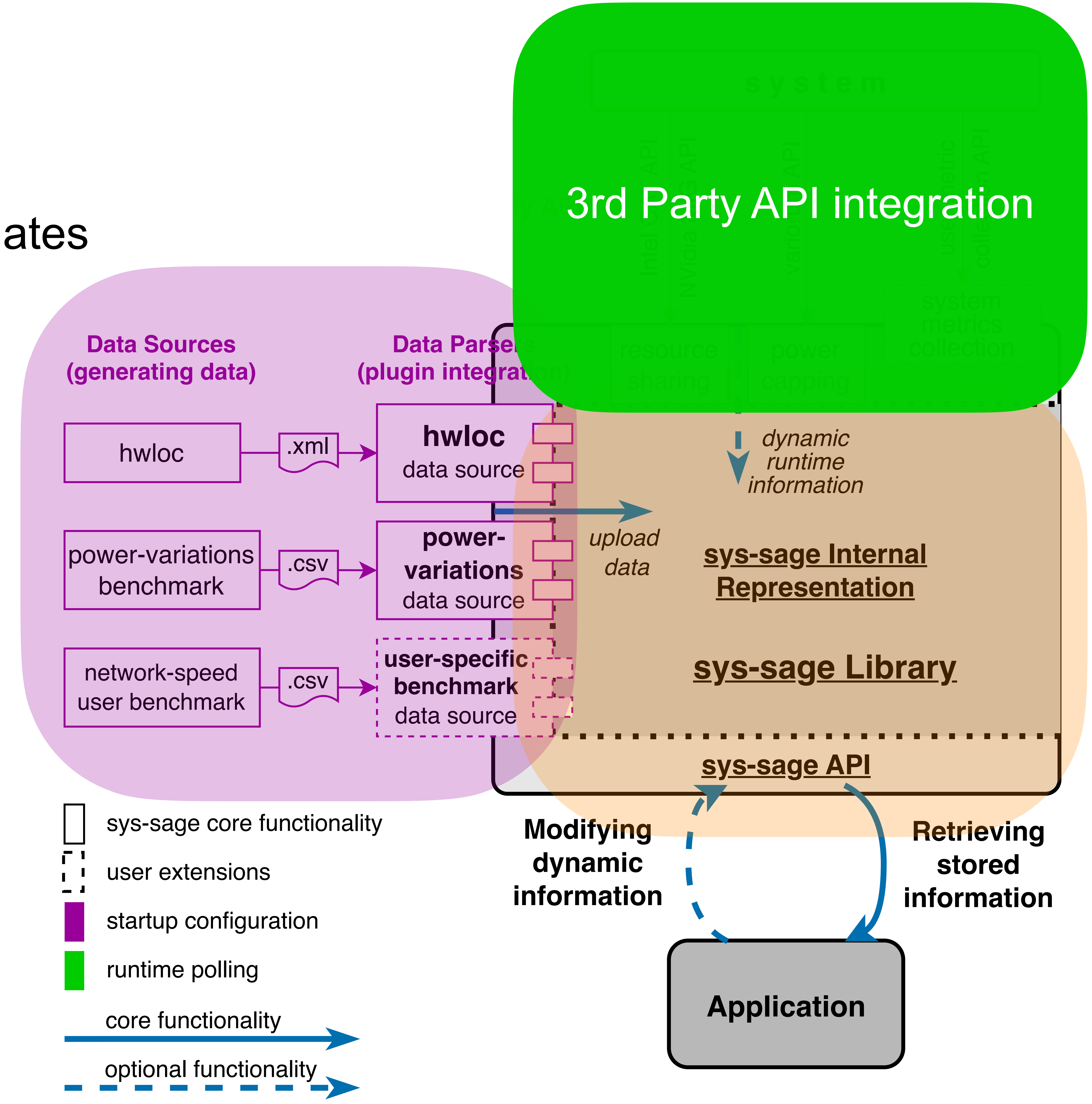
Internal Representation

Stores all the information

- System topologies
- Logical connection of all the data
- Additional attributes

Data Parsers

Interfaces available before startup


3rd Party API integration
Data Sources (generating data)
Data Parsers (plugin integration)
hwloc
.xml
hwloc data source
power-variations benchmark
.csv
power-variations data source
network-speed user benchmark
.csv
user-specific benchmark data source
dynamic runtime information
upload data
sys-sage Internal Representation
sys-sage Library
sys-sage API
Modifying dynamic information
Retrieving stored information
Application
sys-sage core functionality
user extensions
startup configuration
runtime polling
core functionality
optional functionality

Architecture of sys-sage

The Application triggers all data additions/updates

Internal Representation

Stores all the information

- System topologies
- Logical connection of all the data
- Additional attributes

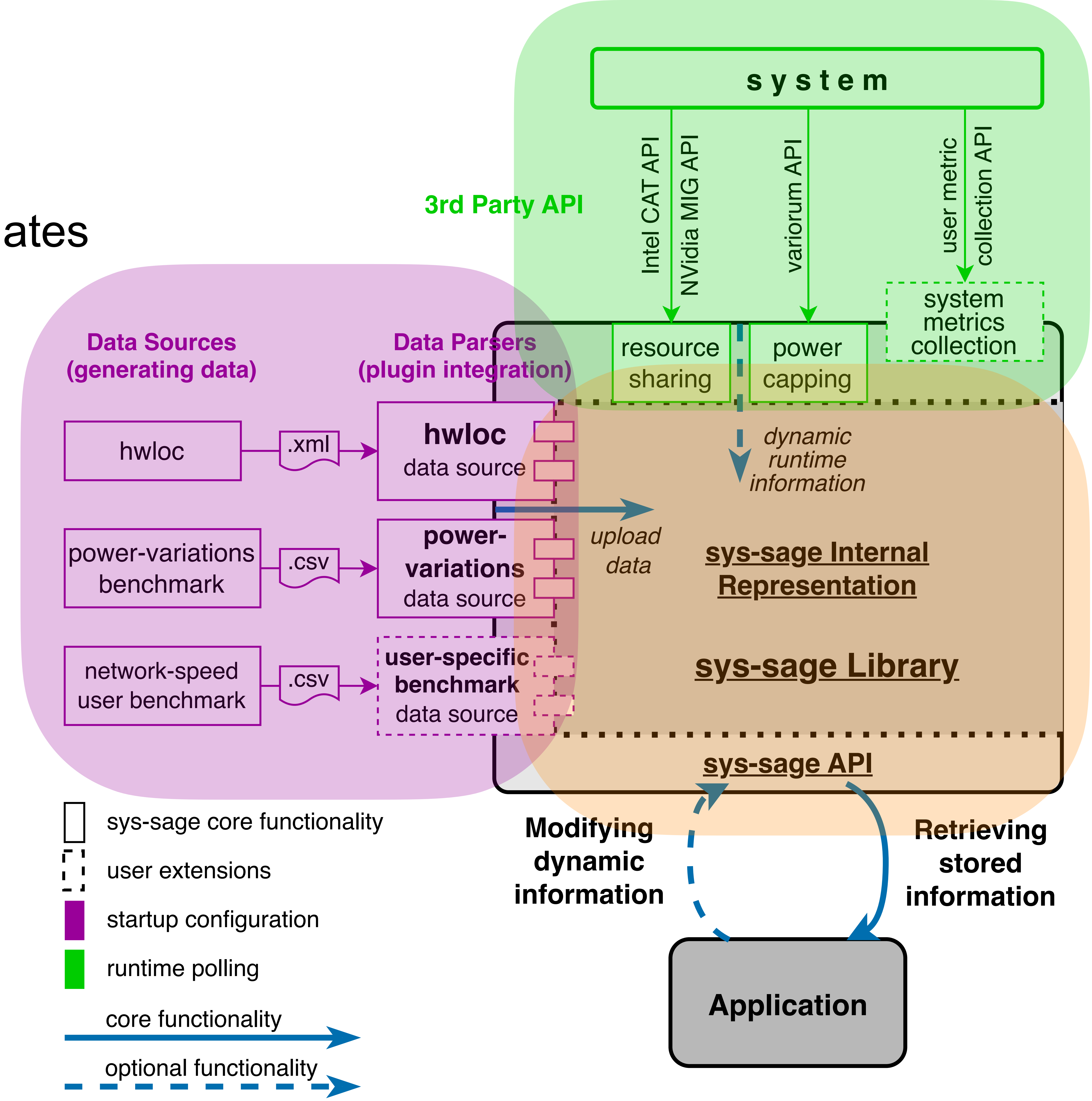
Data Parsers

Interfaces available before startup

3rd Party API Integration

Dynamic data retrieved from 3rd party tools

- Polled once at startup and does not change
- Polled repeatedly dung application lifetime


system
3rd Party API
Intel CAT API
NVidia MIG API
variorum API
user metric collection API
system metrics collection
resource sharing
power capping
Data Sources (generating data)
Data Parsers (plugin integration)
hwloc
.xml
hwloc data source
power-variations benchmark
.csv
power-variations data source
network-speed user benchmark
.csv
user-specific benchmark data source
dynamic runtime information
upload data
sys-sage Internal Representation
sys-sage Library
sys-sage API
Modifying dynamic information
Retrieving stored information
Application
sys-sage core functionality
user extensions
startup configuration
runtime polling
core functionality
optional functionality

Supported integration

Data Parsers

- hwloc (CPU topology)
- mt4g (GPU topology)
- cccbench (CPU core-to-core latencies)
- *(WIP) CPU performance-related benchmarks*

3rd Party Integration

- pqos (Cache partitioning; Intel CPU)
- mig (Resource Isolation; NVidia GPU)
- CPU core frequency
- *(WIP) PAPI (CPU performance counters)*
- *(WIP) variorum (power consumption, power capping)*

…more to come!

sys-sage API

- Functionality for storage, updates and retrieval of the stored information
- Internal representation in form of C++ class objects

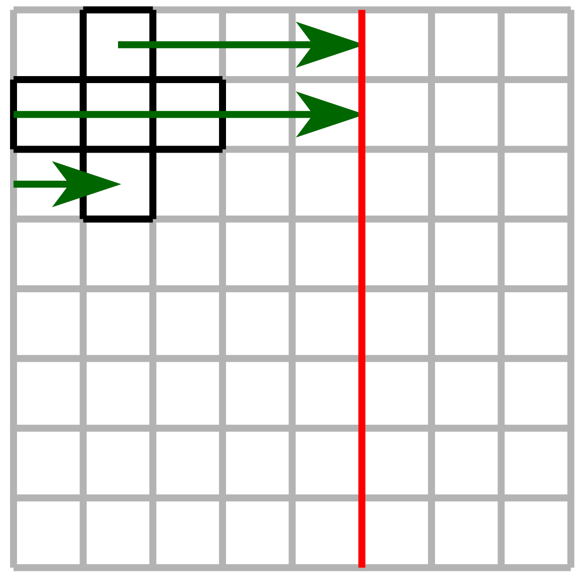
  ➥ **C++ API in form of public class methods**

  ➥ Data Parsers C-style functions
- Full overview available through sys-sage Documentation

Use-case: **Cache Partitioning**

- Cache-aware algorithms tuned to specific cache size
- Cache partitioning enables isolating cache partitions to cores or processes
  - Renders the static (L3) cache size information useless

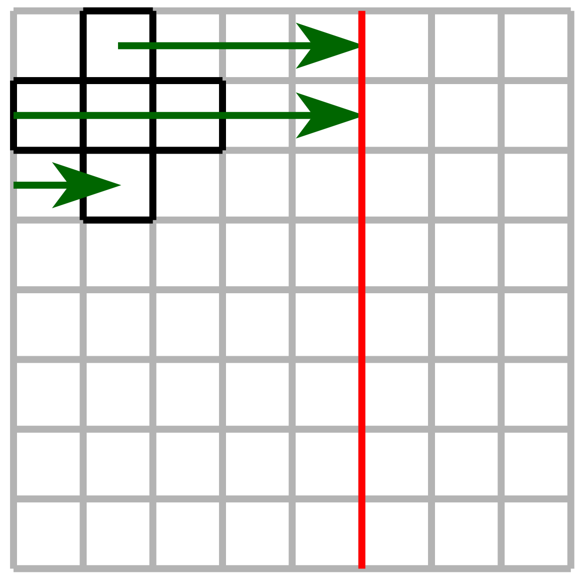
How large blocks?
-> Fit into L3 cache



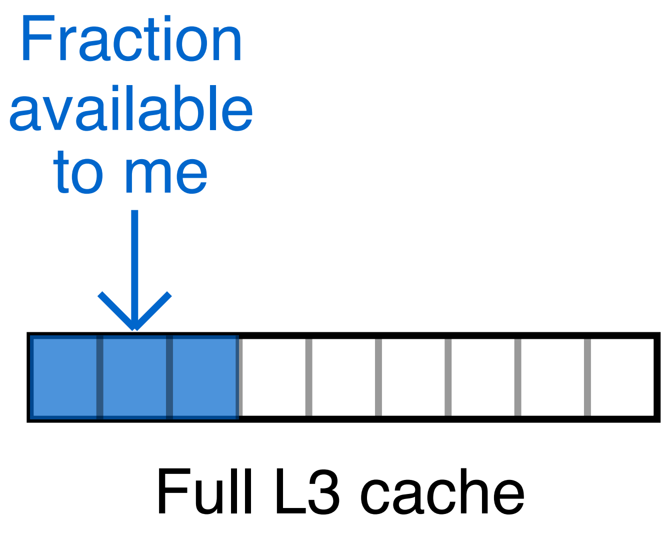
Use-case: **Cache Partitioning**

- Cache-aware algorithms tuned to specific cache size
- Cache partitioning enables isolating cache partitions to cores or processes
    - Renders the static (L3) cache size information useless

How large blocks?
-> Fit into L3 cache



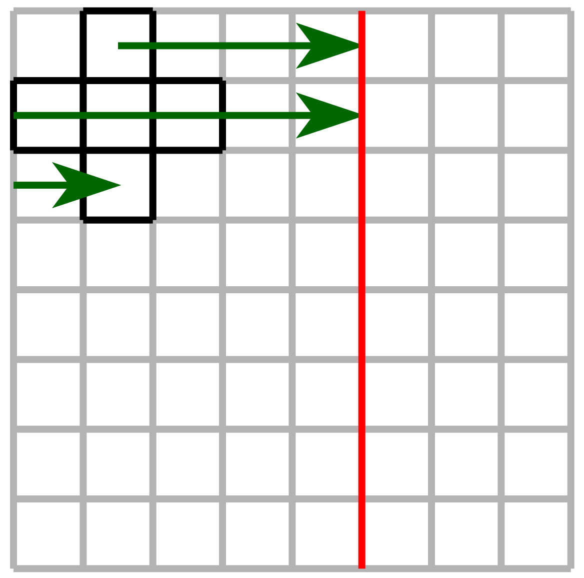
What if L3 cache size changes dynamically?



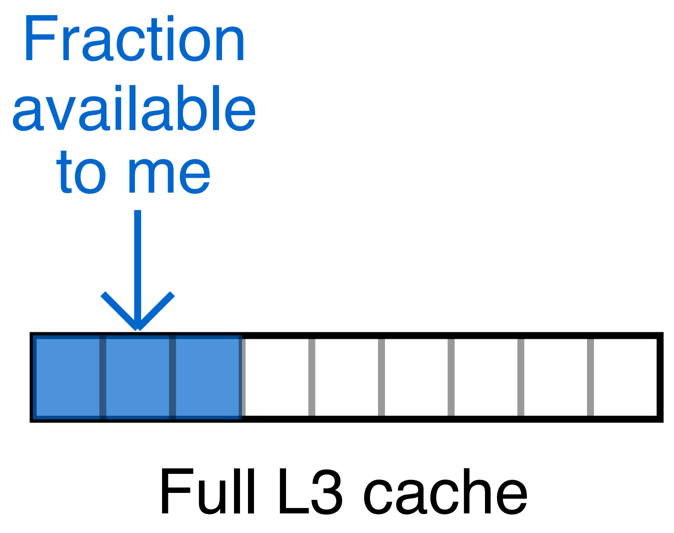
Use-case: **Cache Partitioning**

- Cache-aware algorithms tuned to specific cache size
- Cache partitioning enables isolating cache partitions to cores or processes
  - Renders the static (L3) cache size information useless

How large blocks?
-> Fit into L3 cache



What if L3 cache size changes dynamically?

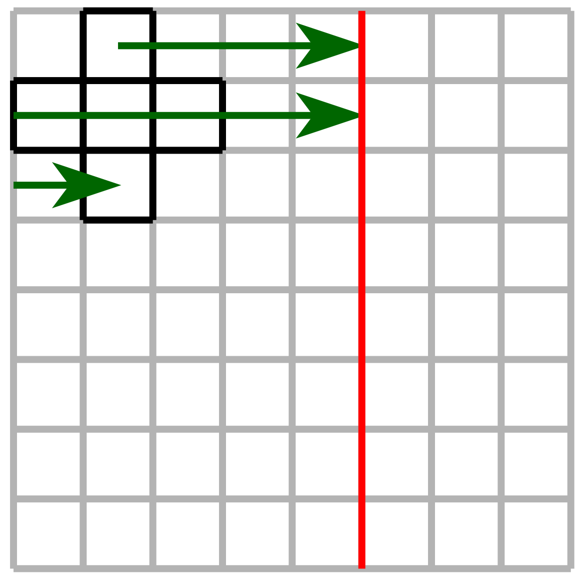


Speedup vs. Naive	No Partitioning	**8/11 L3**	**2/11 L3**
Cache-aware **Static**	1.47	1.20	**1.03**

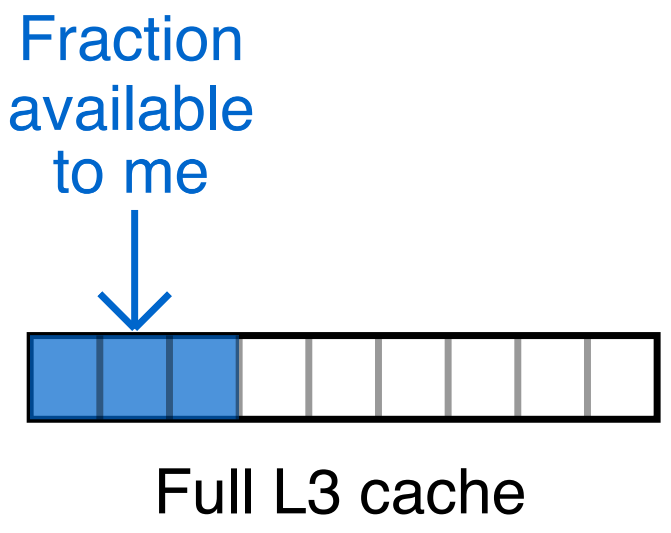
Use-case: **Cache Partitioning**

- Cache-aware algorithms tuned to specific cache size
- Cache partitioning enables isolating cache partitions to cores or processes
  - Renders the static (L3) cache size information useless

How large blocks?
-> Fit into L3 cache



What if L3 cache size changes dynamically?



Static size insufficient

Speedup vs. Naive	No Partitioning	**8/11 L3**	**2/11 L3**
Cache-aware **Static**	1.47	1.20	**1.03**

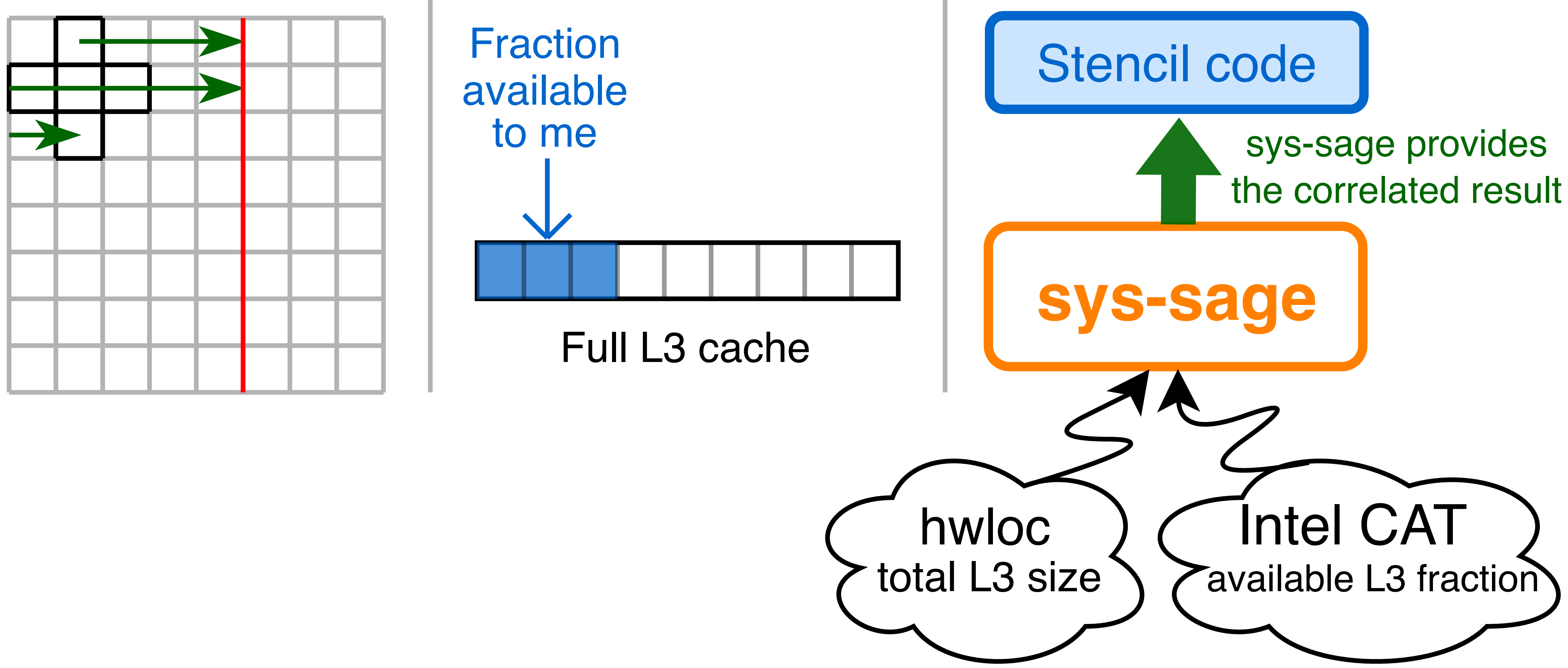
Use-case: **Cache Partitioning**

- Cache-aware algorithms tuned to specific cache size
- Cache partitioning enables isolating cache partitions to cores or processes
  - Renders the static (L3) cache size information useless

How large blocks?
-> Fit into L3 cache

What if L3 cache size changes dynamically?

How to obtain L3 cache size reflecting dynamic system configuration?



Using **sys-sage** to retrieve the **effective L3 cache size**

- Full L3 size and # cache ways (hwloc) in Cache Component
- # open L3 cache ways (pqos) as Data Paths

➥ **calculate available cache size**

Static size insufficient

Speedup vs. Naive	No Partitioning	**8/11 L3**	**2/11 L3**
Cache-aware **Static**	1.47	1.20	**1.03**

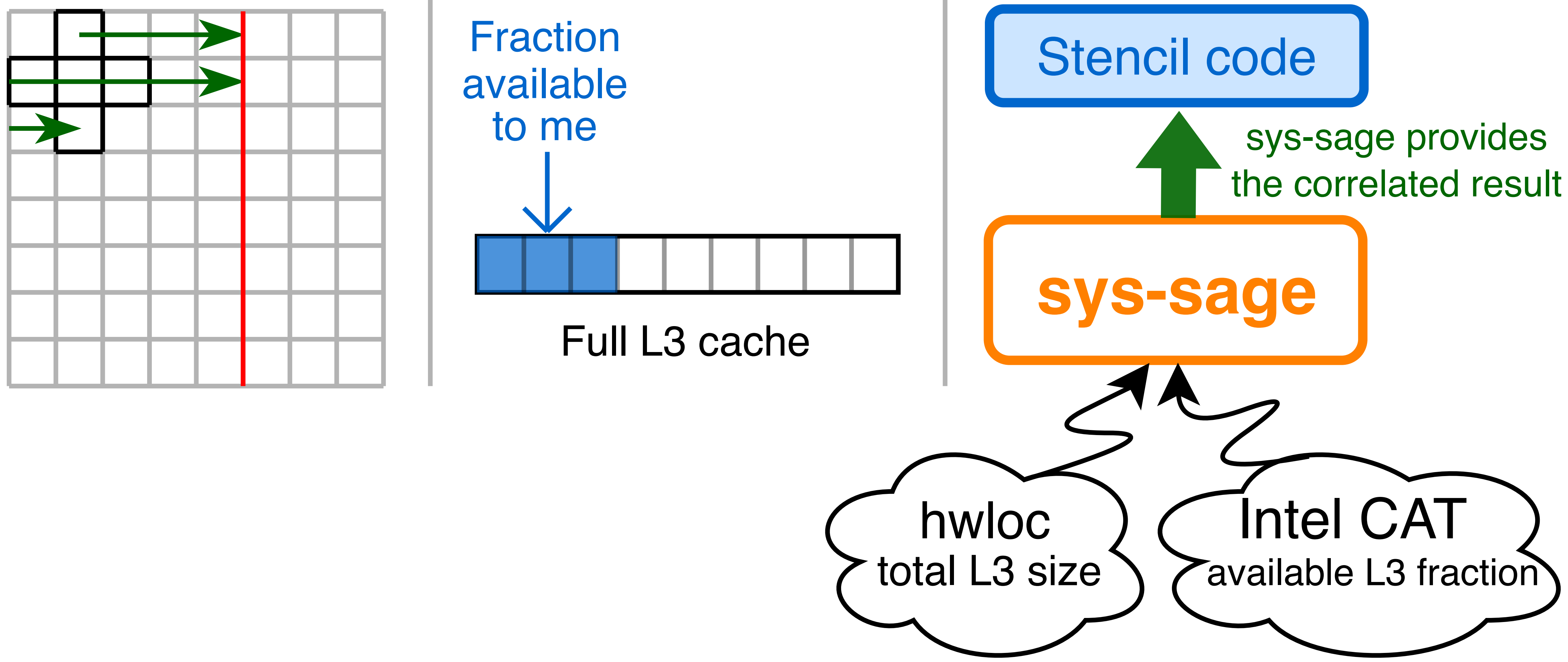
Use-case: **Cache Partitioning**

- Cache-aware algorithms tuned to specific cache size
- Cache partitioning enables isolating cache partitions to cores or processes
  - Renders the static (L3) cache size information useless

How large blocks?
-> Fit into L3 cache

What if L3 cache size changes dynamically?

How to obtain L3 cache size reflecting dynamic system configuration?



Using **sys-sage** to retrieve the **effective L3 cache size**

- Full L3 size and # cache ways (hwloc) in Cache Component
- # open L3 cache ways (pqos) as Data Paths

➥ **calculate available cache size**

Static size insufficient

Speedup vs. Naive	No Partitioning	**8/11 L3**	**2/11 L3**
Cache-aware **Static**	1.47	1.20	**1.03**
Cache-aware **Dynamic (sys-sage)**	1.47	1.43	**2.11**

Use-case: **MemAxes** + Mitos

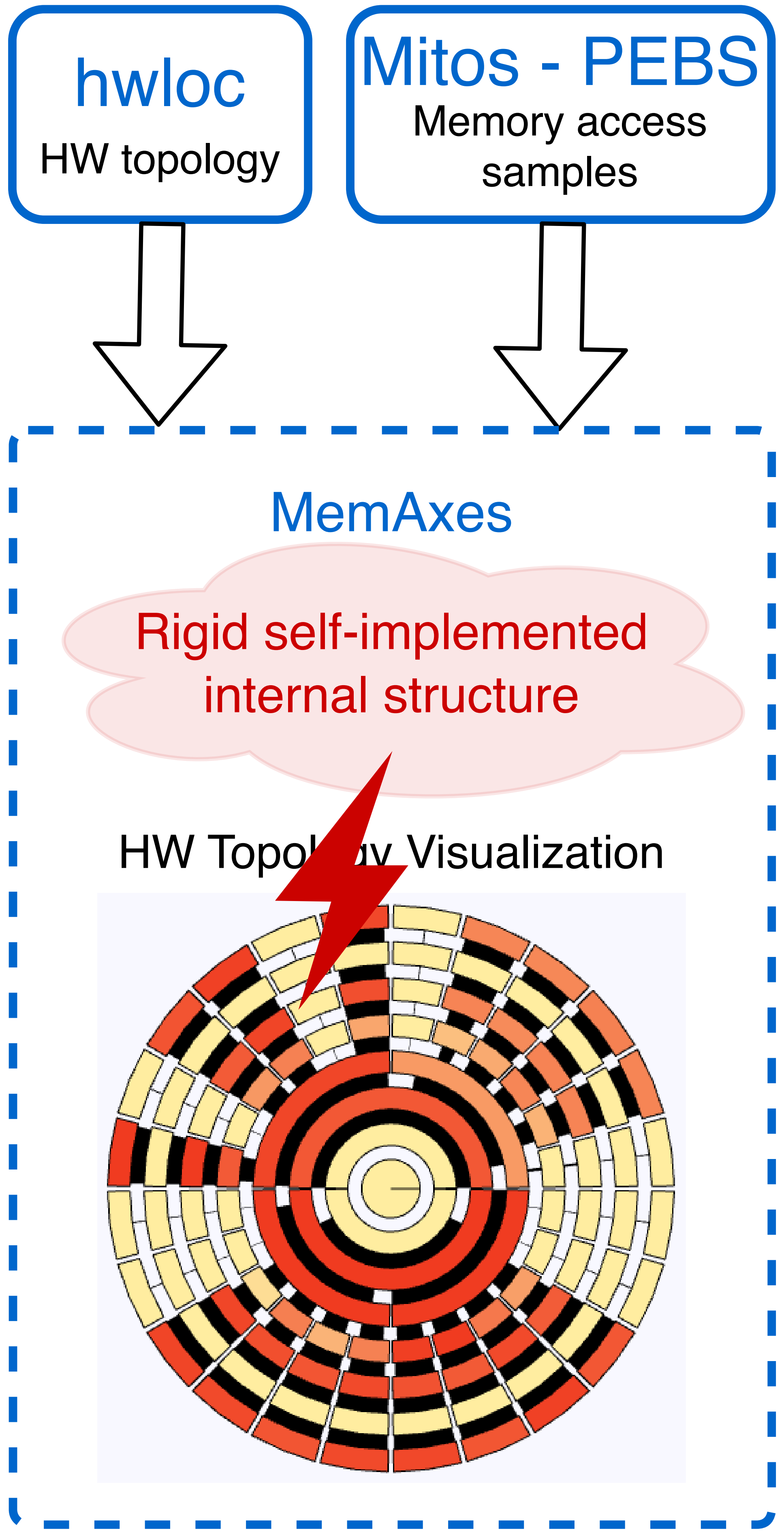
- Mitos: Sample-based data collection
- MemAxes: Visualizing data access characteristics
- Attributing samples to HW resources and source code

Integration with sys-sage

- HW topology (**hwloc**) as **Component Tree**
- **Mitos** samples as **Data Paths**

Benefits

- Flexible and future-proof for modern architectures
- Representing cross-NUMA data accesses
- Extension to AMD IBS samples
- Integration of MUSA simulator


hwloc
HW topology
Mitos - PEBS
Memory access samples
MemAxes
Rigid self-implemented internal structure
HW Topology Visualization

Use-case: **MemAxes** + Mitos

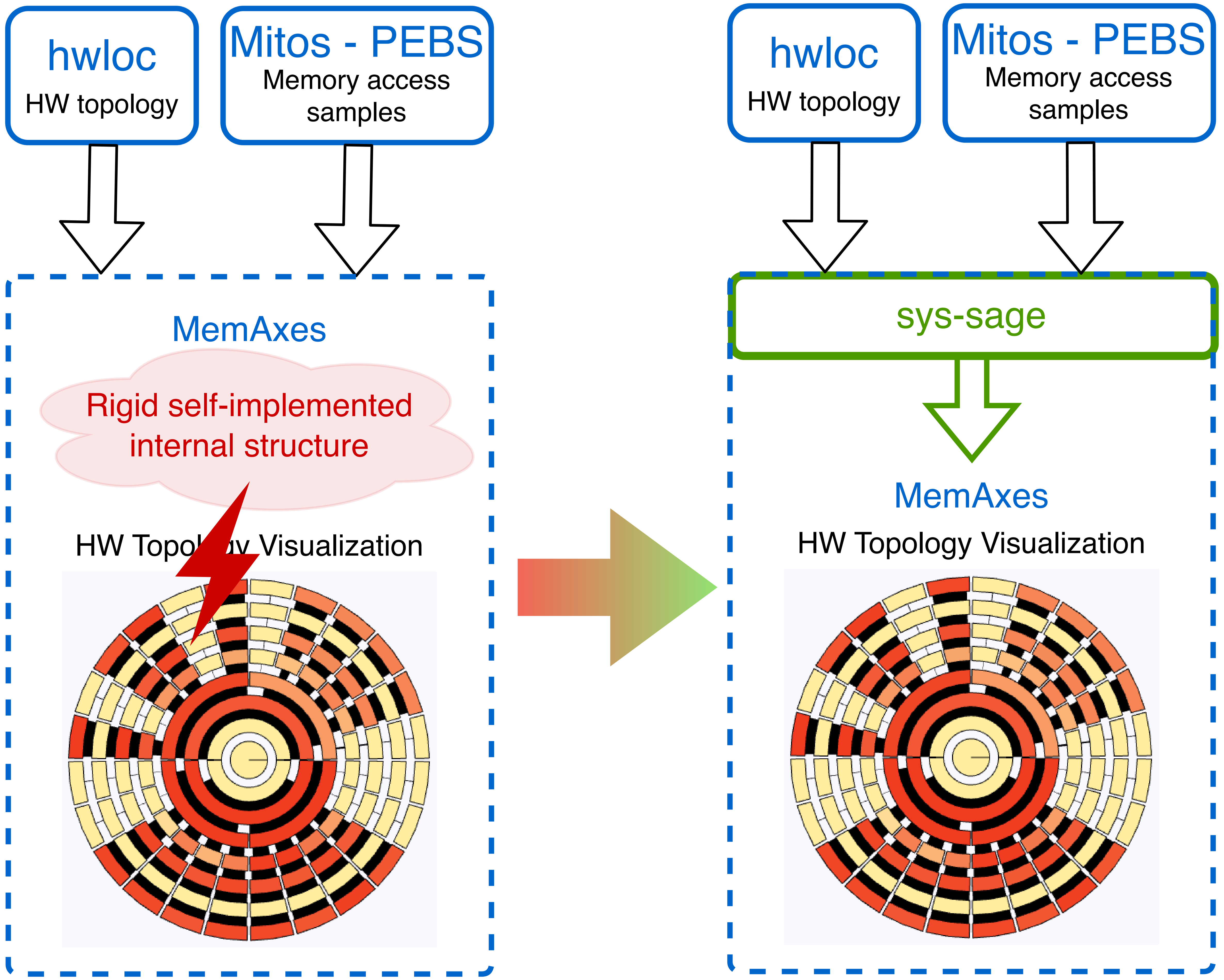
- Mitos: Sample-based data collection
- MemAxes: Visualizing data access characteristics
- Attributing samples to HW resources and source code

Integration with sys-sage

- HW topology (**hwloc**) as **Component Tree**
- **Mitos** samples as **Data Paths**

Benefits

- Flexible and future-proof for modern architectures
- Representing cross-NUMA data accesses
- Extension to AMD IBS samples
- Integration of MUSA simulator



hwloc
HW topology

Mitos - PEBS
Memory access samples

MemAxes

Rigid self-implemented internal structure

HW Topology Visualization

hwloc
HW topology

Mitos - PEBS
Memory access samples

sys-sage

MemAxes

HW Topology Visualization

Use-case: **MemAxes** + Mitos

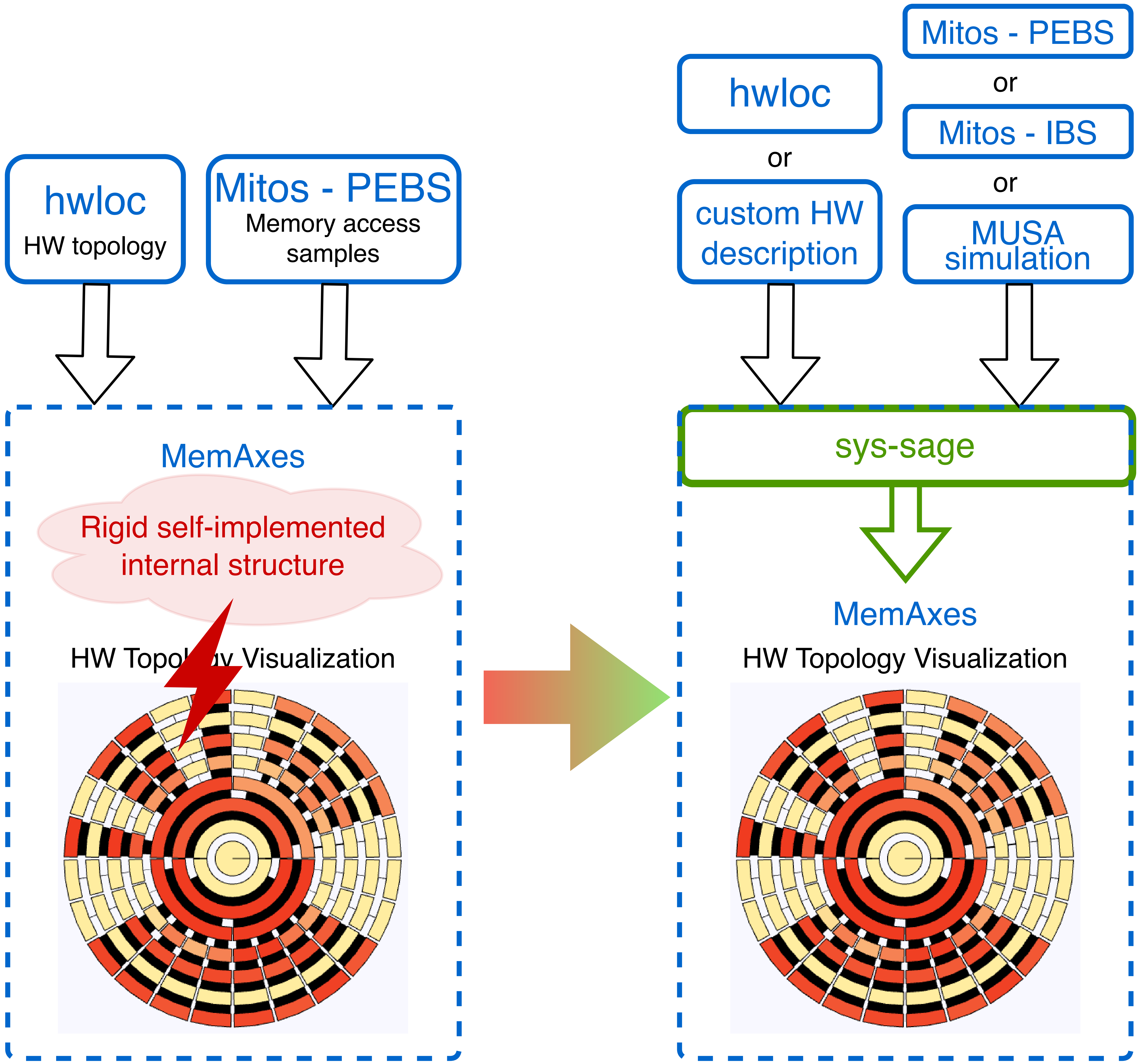
- Mitos: Sample-based data collection
- MemAxes: Visualizing data access characteristics
- Attributing samples to HW resources and source code

Integration with sys-sage

- HW topology (**hwloc**) as **Component Tree**
- **Mitos** samples as **Data Paths**

Benefits

- Flexible and future-proof for modern architectures
- Representing cross-NUMA data accesses
- Extension to AMD IBS samples
- Integration of MUSA simulator



sys-sage

- A C++ based library for managing and representing HPC system topology information
- Unifies different data sources providing partial information
- Integrates multiple popular interfaces
- Open source

Try out sys-sage and get in touch with us!

https://github.com/caps-tum/sys-sage

spack install sys-sage

stepan.vanecek@tum.de

Acknowledgements

SPONSORED BY THE

Federal Ministry
of Education
and Research

DEEP-SEA
EU Grant #955606
BMBF #16HPC014

DEEP-SEA