

CLARA: A Contemporary Approach to Physics Data Processing

V Gyurjyan, D Abbott, J Carbonneau, G Gilfoyle,
D Heddle, G Heyes, S Paul, C Timmer, D Weygand

Problem Statement

- Large software base.
- Dynamic author base.
 - To complete quality physics data production, intellectual input from far-flung groups must be brought together.
- Long life time.
 - The relatively short amount of time spent on experimental data acquisition requires a comparatively long time on data analysis.

Dream vs. Reality



Complex vs. Complicated

	GST definition	Behavior	
	Simple	Easily knowable	
	Complicated	not simple, but still knowable	
	Complex	not fully knowable, but reasonably predictable	
	Chaotic	neither knowable nor predictable	

“In a large software application there are many, densely interconnected parts which affect the behavior of the whole in a non-linear way that cannot be simply understood by looking at the components in isolation.”

Thomas Homer-Dixon

Monolithic vs. Service Oriented



CIARA

- SOA based physics data production application development framework, written in pure Java.
- Service development environment (services in C, C++, Python, Java).
- Increase intrinsic interoperability of services by standardizing data exchange interface.
- Increase federation.
 - Services and ClaRA based applications are united while maintaining their individual autonomy and self governance.
- Multi-Threaded event processing.
- Distributed event processing.
- Ease of application deployment.
- PDP application diversification and agility.
- Clear separation between PDP application designer and service programmer.
 - Build and run PDP applications without an access to the source code of individual services.

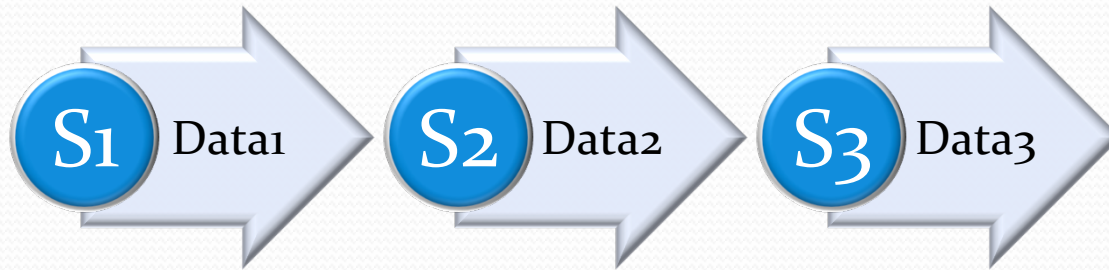


SOA Defined

- An architecture (NOT a technology) based on well defined, reusable components: services.
- Services are loosely coupled.
- Services with functional context that are agnostic to a composed application logic.
- Agnostic services participate in multiple algorithmic compositions.
- Application design based on available services.
- Service location is transparent.
- Service is defined by the messages it can accept and the responses it can give.
- Service implements standard contract/interface.

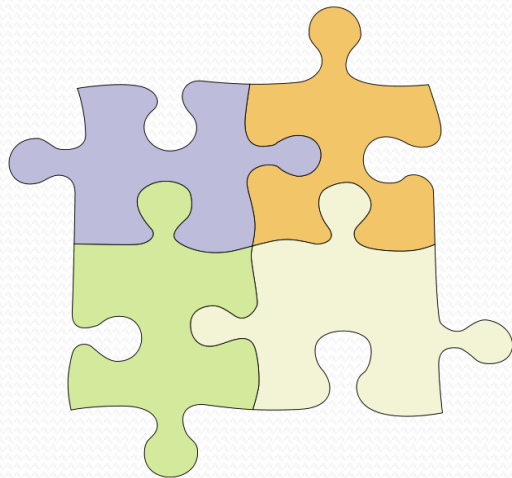
PDP Application Design Data Centric Approach

- Focus on data that is moving and transforming in the system.
- Data flow defines the essential aspect of an application.



Object Centric	Data Centric
Data encapsulation	Data exposure
Object/method exposure	Object/method encapsulation
Intermix of data and algorithm	Separate data and algorithm
Tightly coupled	Loosely coupled

Programmer



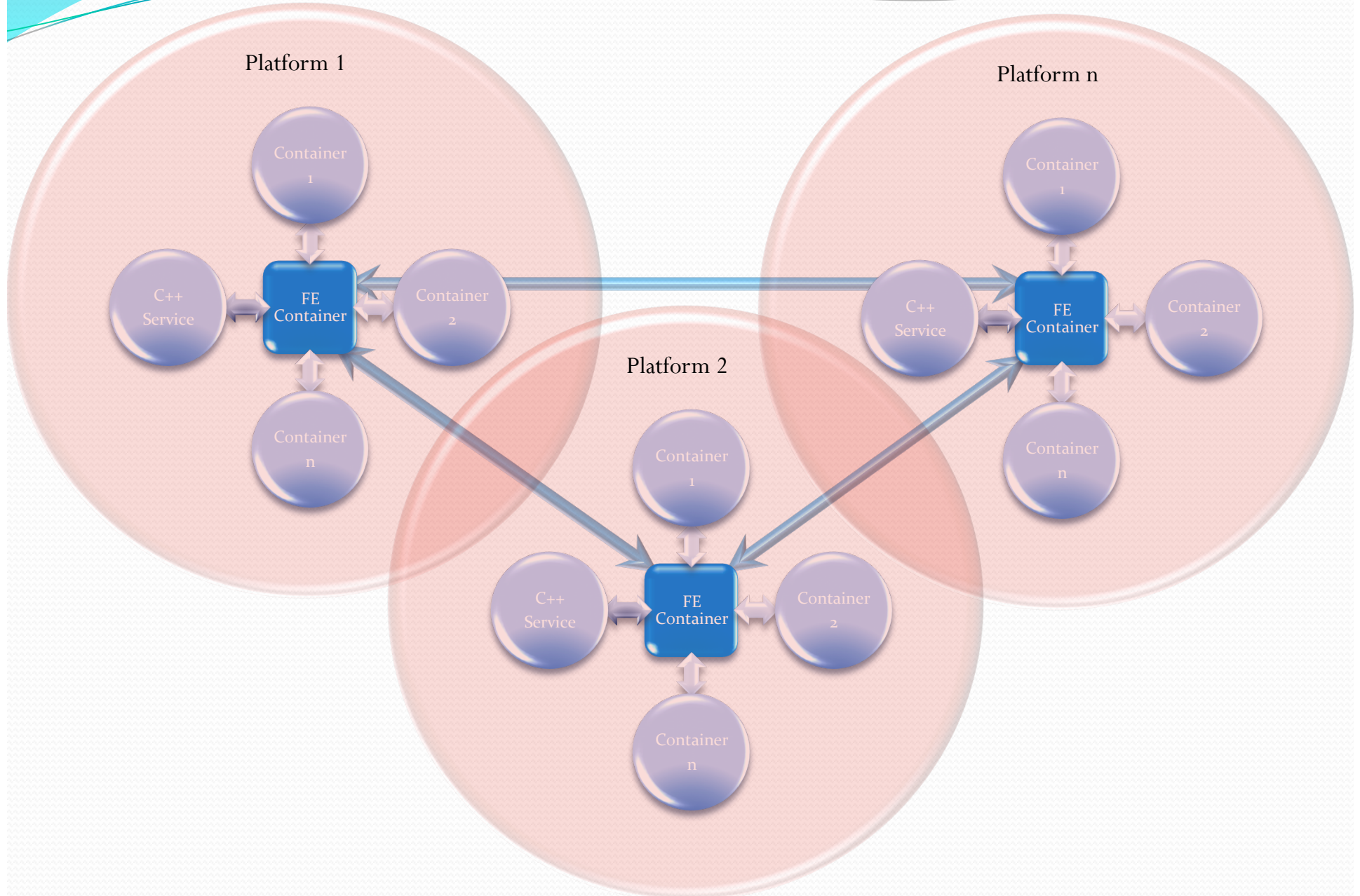
Services

Designer
without programming skills

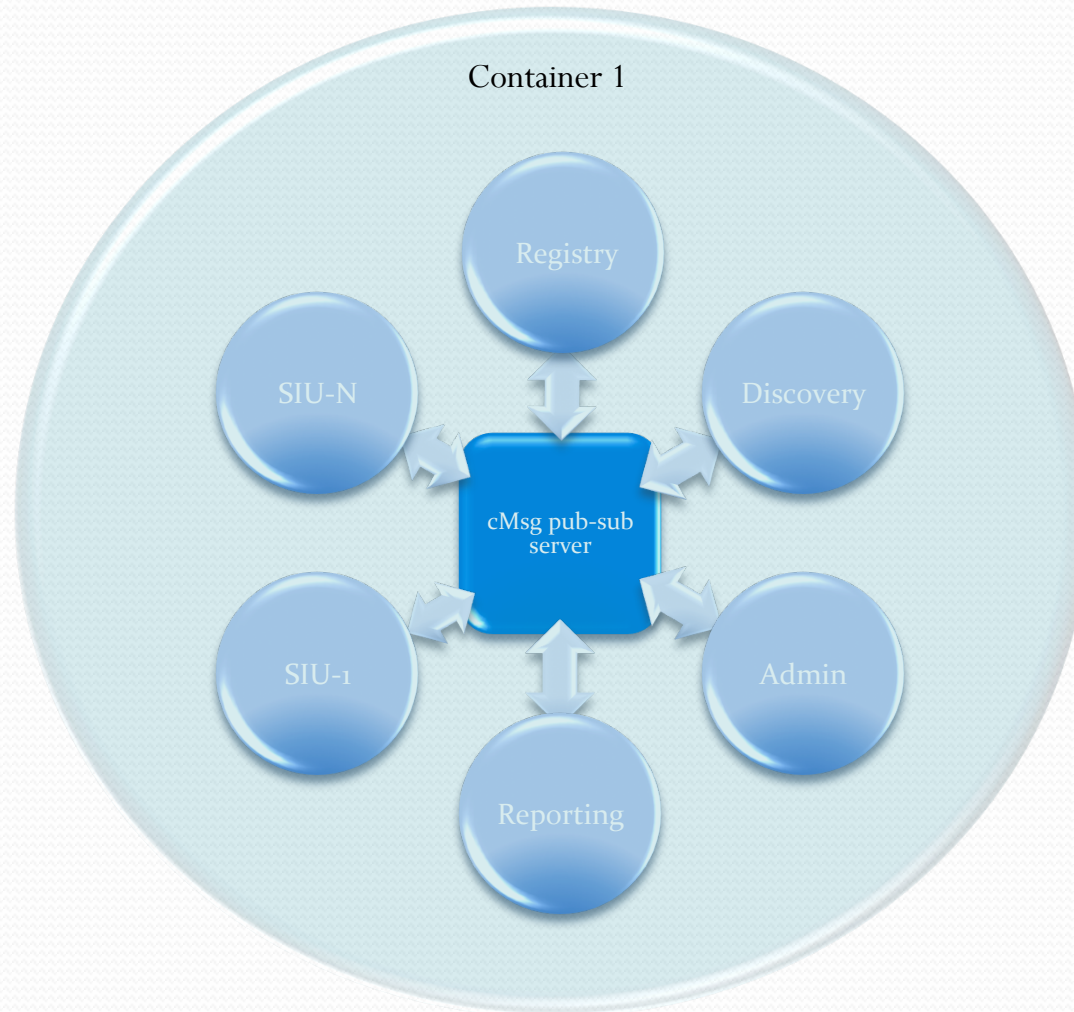


PDP Applications

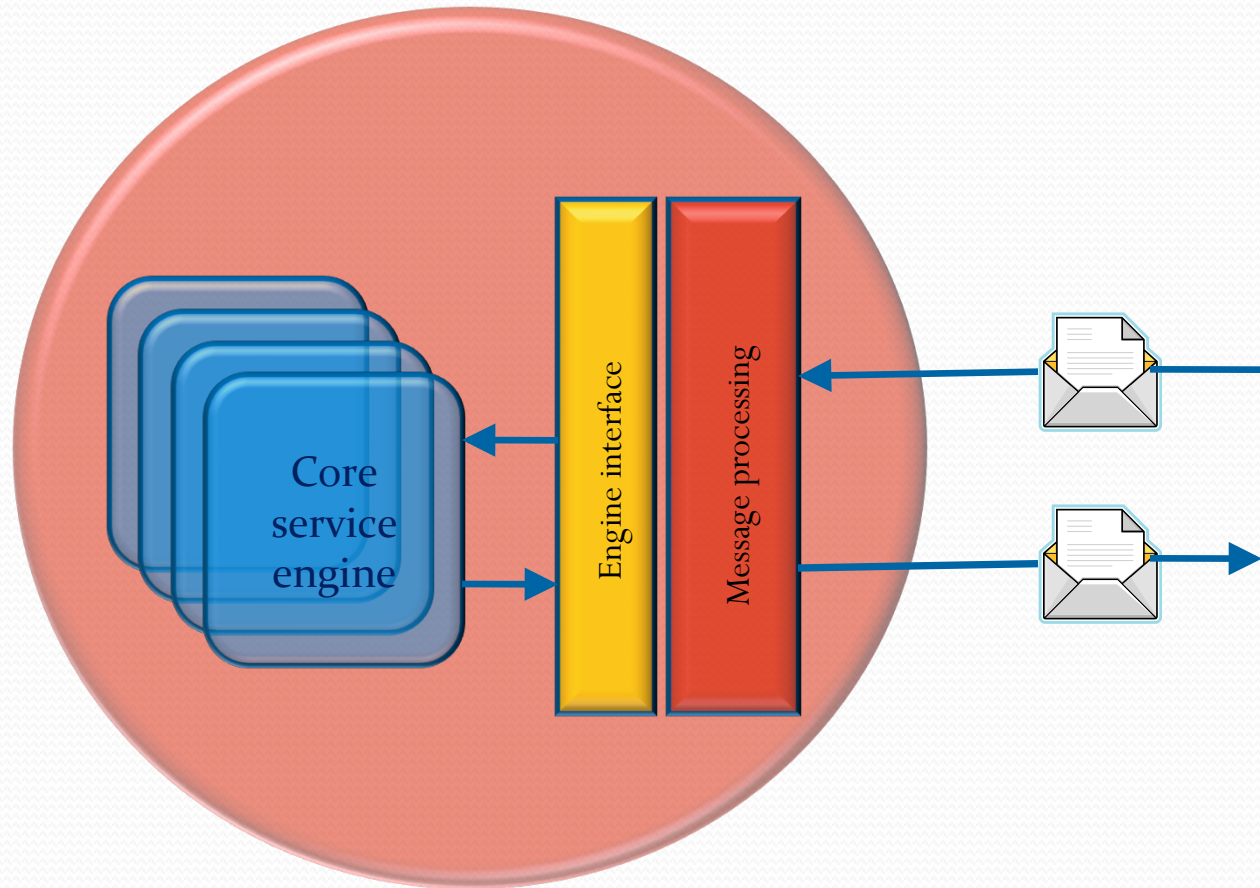
Cloud of CIARA Platforms



CIARA Container



SIU: CIARA Service Engine Integration Unite

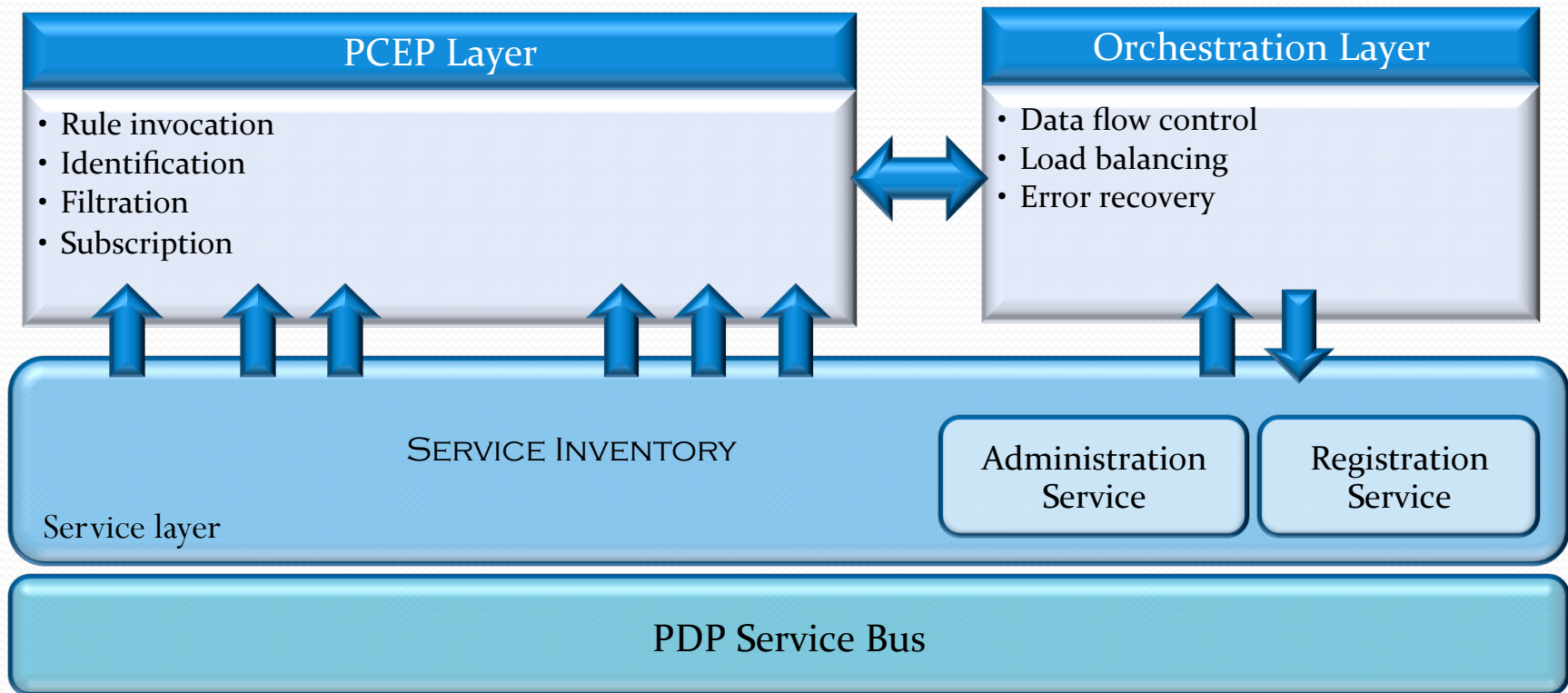


The key is - Standard & Simple Interface



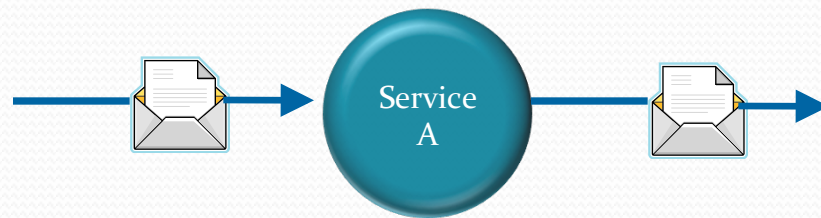
	Type	Definition
Name	String	Service name
Description	String	Service description
Author	String	Service author
Version	String	Service version
Input Data Type	EVIO, primitives	Accepted
Output Data Type	EVIO, primitives	Generated

ClaRA Design Architecture



Service Coupling

- A functional service context is independent from the outside logic.
- Contract-To-Functional coupling between services.
 - Service has a name.
 - Service receives data and sends data.
- Consumer-To-Contract coupling between consumers and services.
 - User/consumer sends data to the named service.
 - Sending the data will trigger execution of the receivers service.





Service abstraction

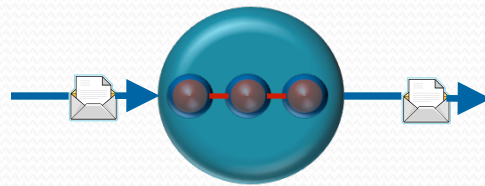
- Technology information (hidden)
- Programmatic logic information (hidden)
- Functional information (exposed through service contract meta data)
- Quality of service information (available from the platform registration services)

Service types

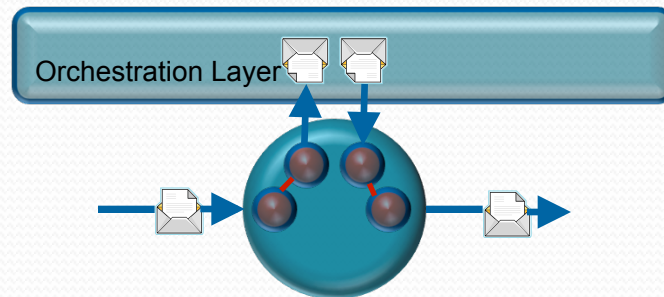
- Entity services
 - Generic
 - Highly reusable
- Utility services
 - Self contained
 - Legacy systems
- Composite services
 - Primitive compositions
 - Self governing service compositions, task services
 - Complex compositions
 - Controlled aggregate services

Service Composition

- Primitive composition
 - Represents message exchanges across two or more services.
 - Requires composition initiator.
 - Task services are examples of primitive composition.

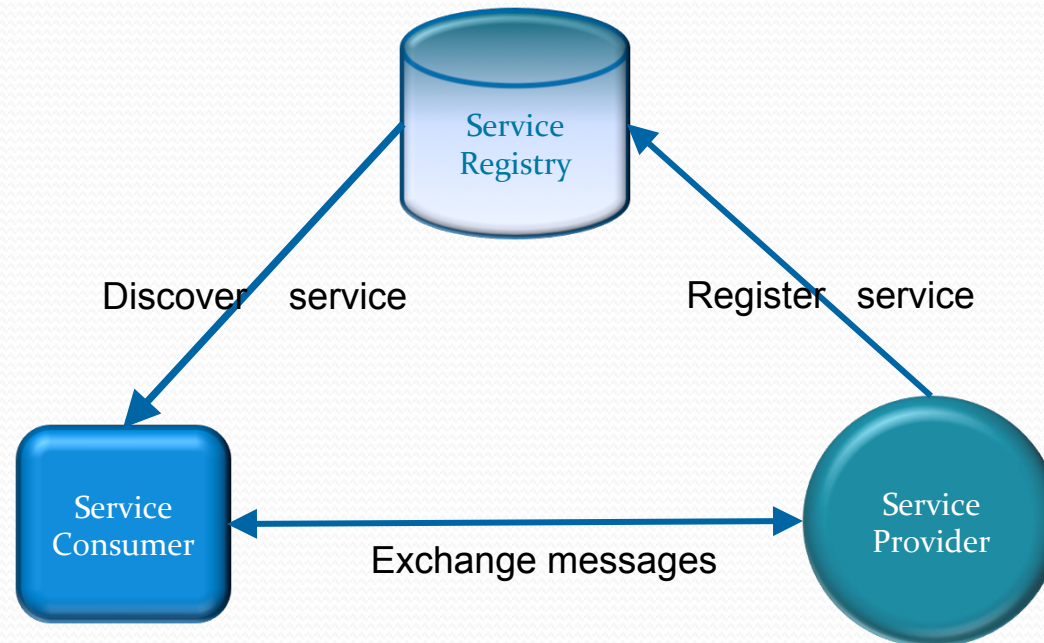


- Complex composition
 - Orchestrated task services.



Service Discovery

- Design time
- Runtime

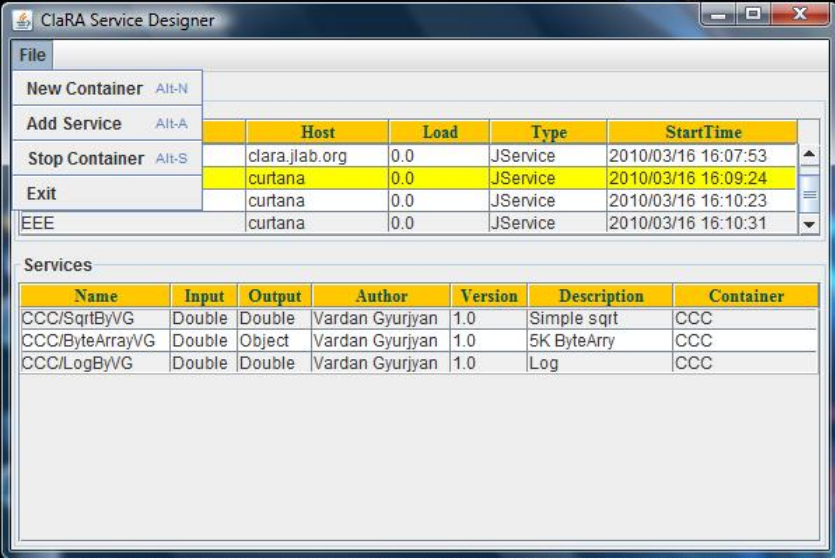


Scaling

- Only one process is active on the CLARA platform node (CLARA container).
- Single CLARA container (JVM) on a node.
- Service containers run in their own threads.
- A single service container executes contained service engines in separate threads.
- A service engine must be thread safe if it is planned to run in a multi-threaded mode.
- CLARA based PDP application gets inherent multi-threading and distributed processing, with the estimated relative processing time :

$$\Delta T_R = (\Delta T_A + \Delta x_n) / N_c * N_n$$

Service deployment and monitoring Interfaces



The screenshot shows the 'ClaRA Service Designer' application window. It features a menu bar with 'File' and several options: 'New Container' (Alt+N), 'Add Service' (Alt+A), 'Stop Container' (Alt+S), and 'Exit'. Below the menu is a table with columns: Host, Load, Type, and StartTime. The table contains four rows of data, with the second and third rows highlighted in yellow. Below this is a 'Services' section with a table containing columns: Name, Input, Output, Author, Version, Description, and Container. This table lists three services: CCC/SqrtByVG, CCC/ByteArrayVG, and CCC/LogByVG.

	Host	Load	Type	StartTime
	clara.jlab.org	0.0	JService	2010/03/16 16:07:53
	curtana	0.0	JService	2010/03/16 16:09:24
	curtana	0.0	JService	2010/03/16 16:10:23
EEE	curtana	0.0	JService	2010/03/16 16:10:31

Name	Input	Output	Author	Version	Description	Container
CCC/SqrtByVG	Double	Double	Vardan Gyurjyan	1.0	Simple sqrt	CCC
CCC/ByteArrayVG	Double	Object	Vardan Gyurjyan	1.0	5K ByteArray	CCC
CCC/LogByVG	Double	Double	Vardan Gyurjyan	1.0	Log	CCC

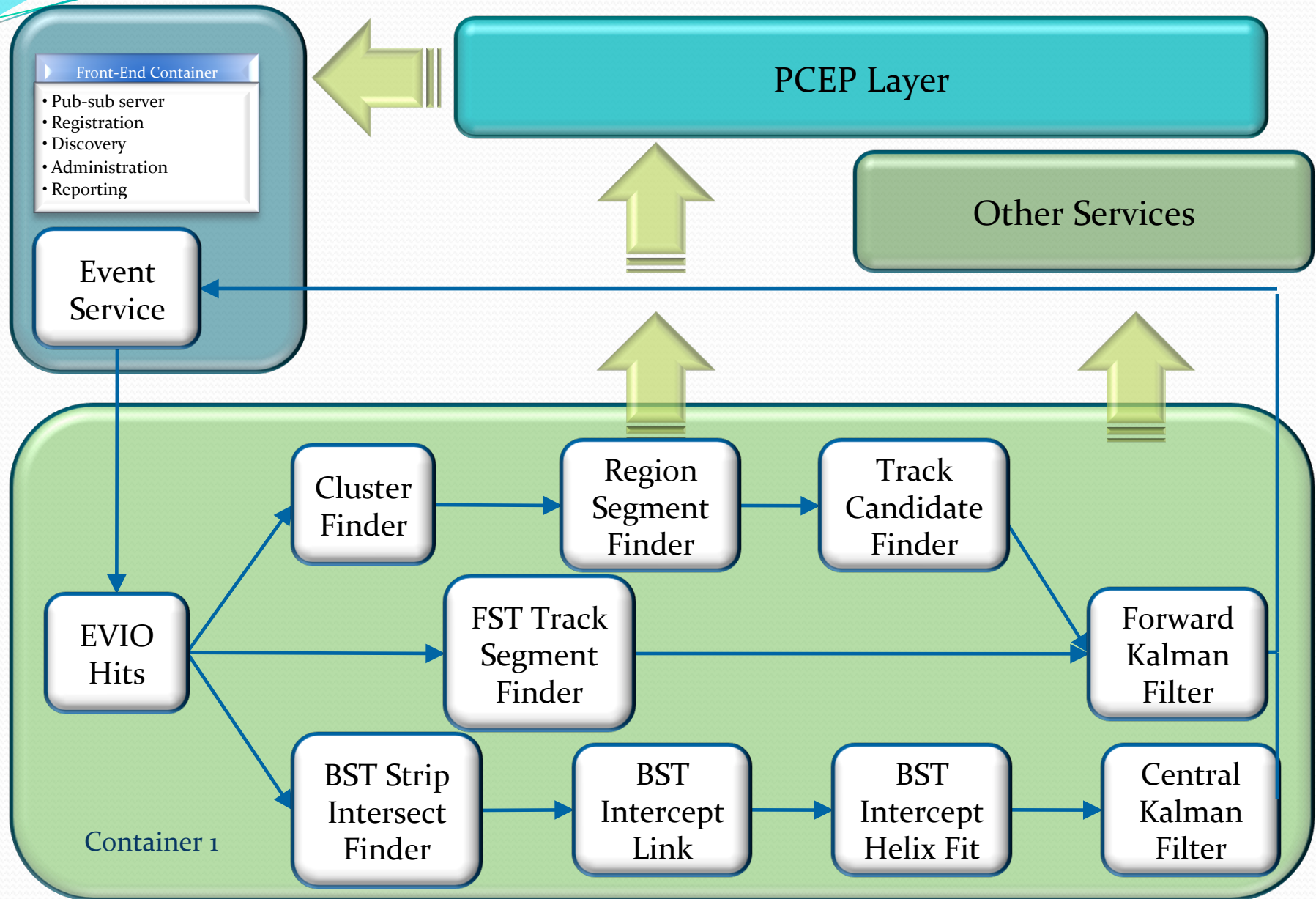


CLAS12 SOT: Service Oriented Tracking

SOT Service Inventory

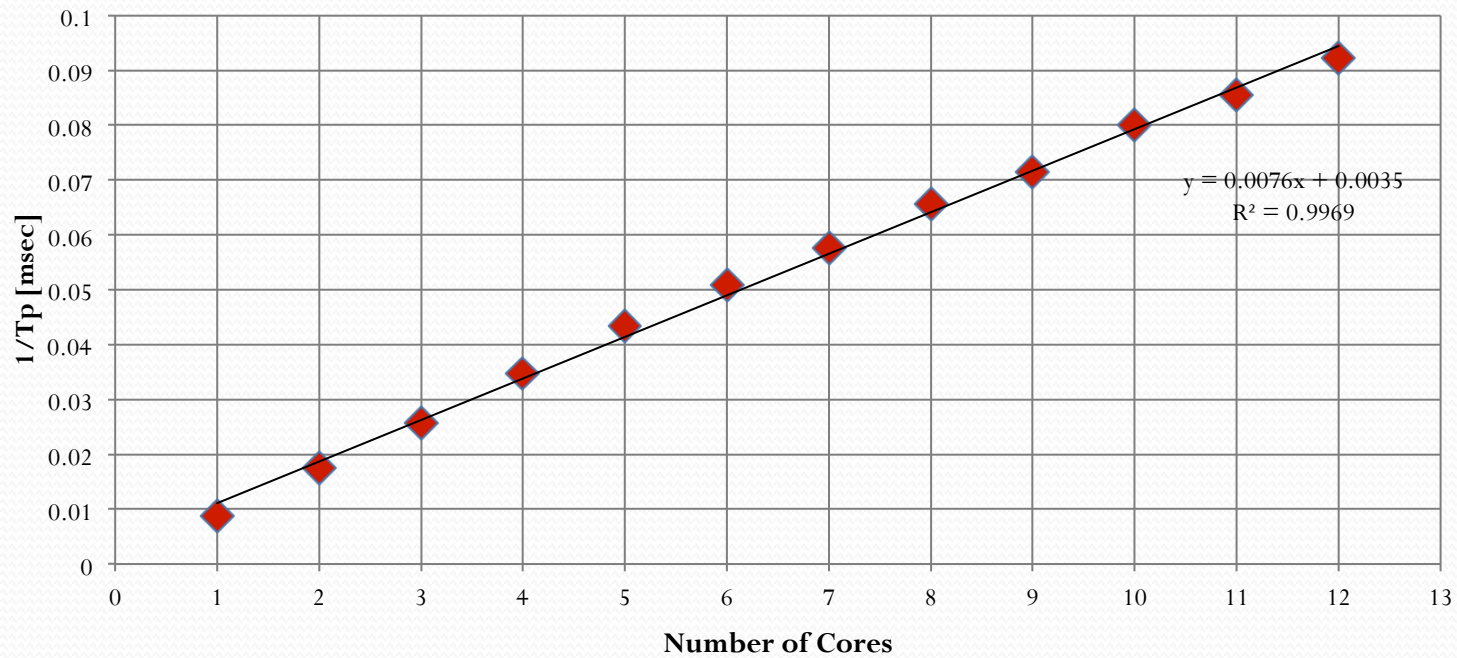
Service	Detector
Event Service	n/a
Cluster Finder	DC
Region Segment Finder	DC
Track Candidate Finder	DC
Track Segment Finder	FST
Strip Intersect Finder	BST
Intercept Linker	BST
Intercept Helix Fit	BST
Kalman Filter	Forward Detectors
Kalman Filter	Central Detectors
DC Geometry	DC
FSC Geometry	FSC
BST Geometry	BST

Clas12 Tracking Application Design



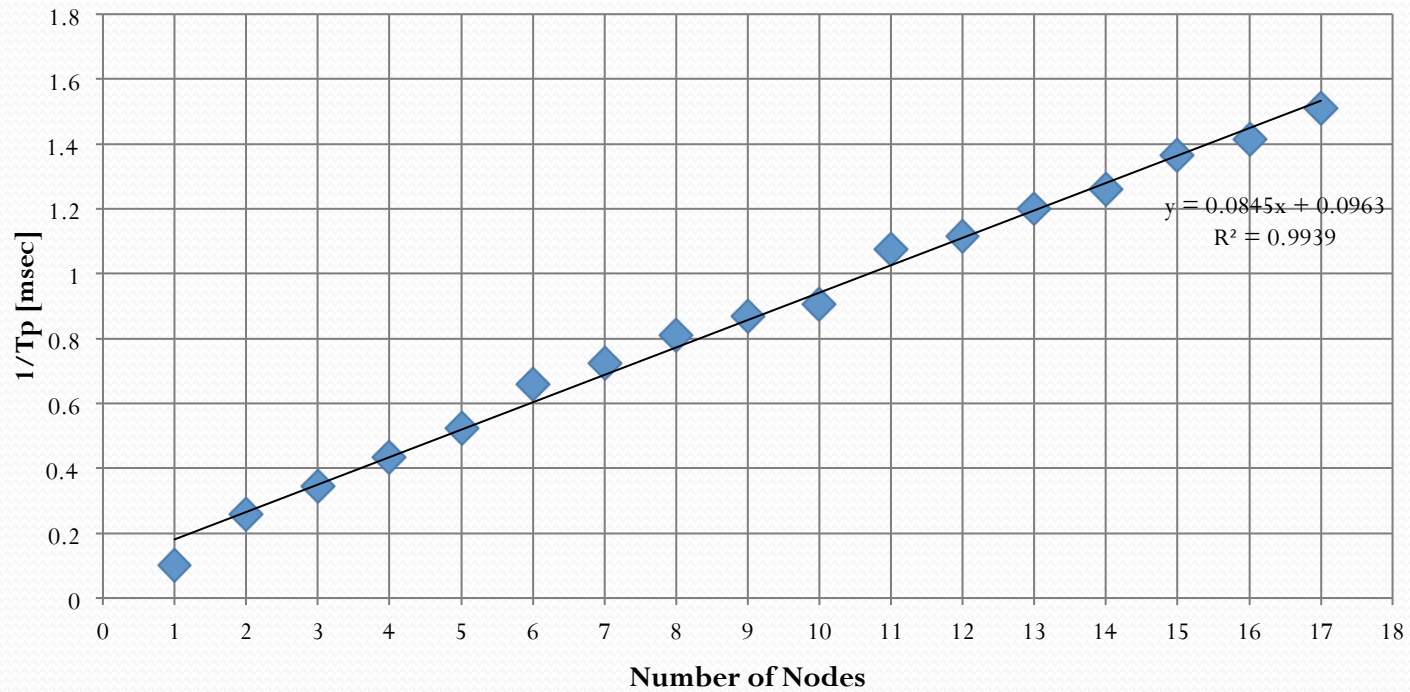
Composite Reconstruction Service Multi-Threading

**CLARA SOT performance in an Intel Xeon 2x6 Westmere
processor based node**



Composite Reconstruction Service Distributed Processing

CLARA SOT performance in a cluster



Conclusion

- SOA based physics data production application development framework, written in pure Java.
- Separation between PDP application designer and service programmer.
- Service development environment.
- Increase intrinsic interoperability of services by standardizing data exchange interface.
- Increase federation.
- Multi-Threaded event processing.
- Distributed event processing.
- Ease of application deployment.
- Increase application diversification and agility.
- Designed and deployed service based Clas12 full track reconstruction application, showing ~650 micro second per event relative processing time on the ClaRA platform using 17 Xeon 2x6 Westmere nodes.