



## A Simple CCA Component Application

**CCA Forum Tutorial Working Group**  
<http://www.cca-forum.org/tutorials/>



## Module Overview

- What the example does: the math.
- From math to components: the architecture.
- The making of components: inheritance and ports.
- Framework-component interactions.
- Putting it all together: the Ccafeine ways.
- The application in action.

## Goals

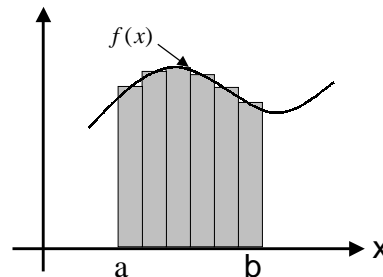
To show how CCA components are used to build an application to numerically integrate a continuous function using two different integration techniques.

3

## The Math: Integrator (1)

### The midpoint numerical integrator

$$\int_a^b f(x) dx \approx \frac{b-a}{n} \sum_{j=1}^n f\left(\frac{x_{j-1} + x_j}{2}\right)$$



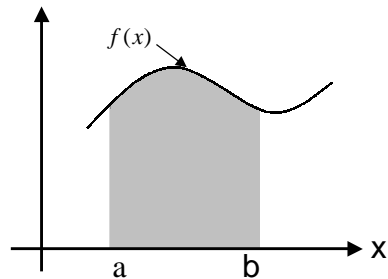
4

## The Math: Integrator (2)

### The Monte Carlo integrator

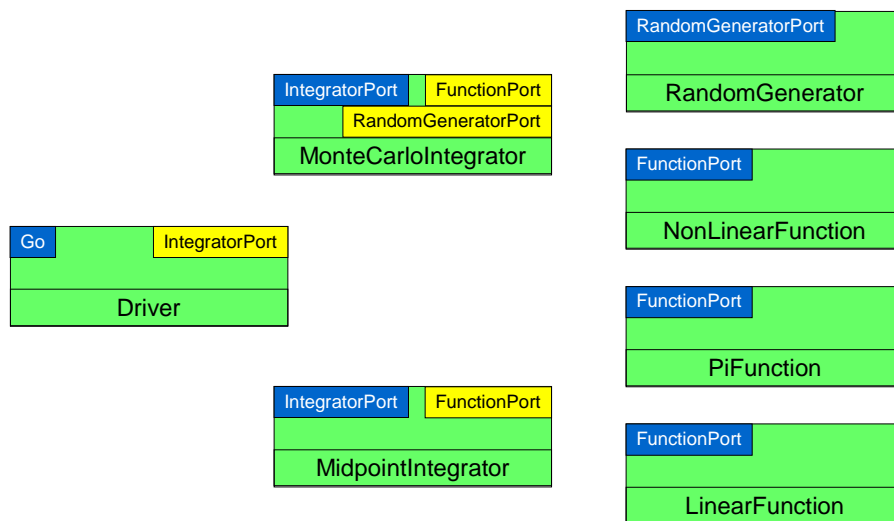
$$\int_a^b f(x)dx \approx \frac{1}{b-a} \left( \frac{1}{N} \sum_{i=1}^N f(x_n) \right)$$

$x_n$  Uniformly distributed in  $[a, b]$



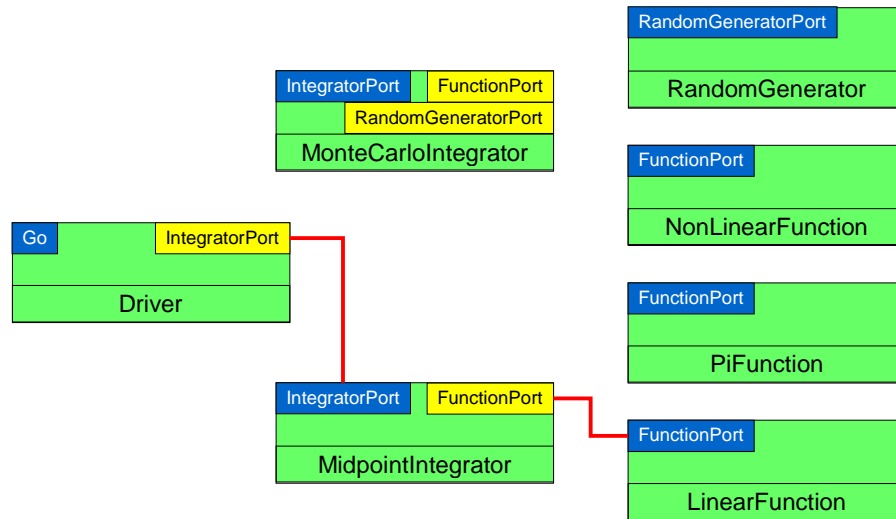
5

## Example Architecture



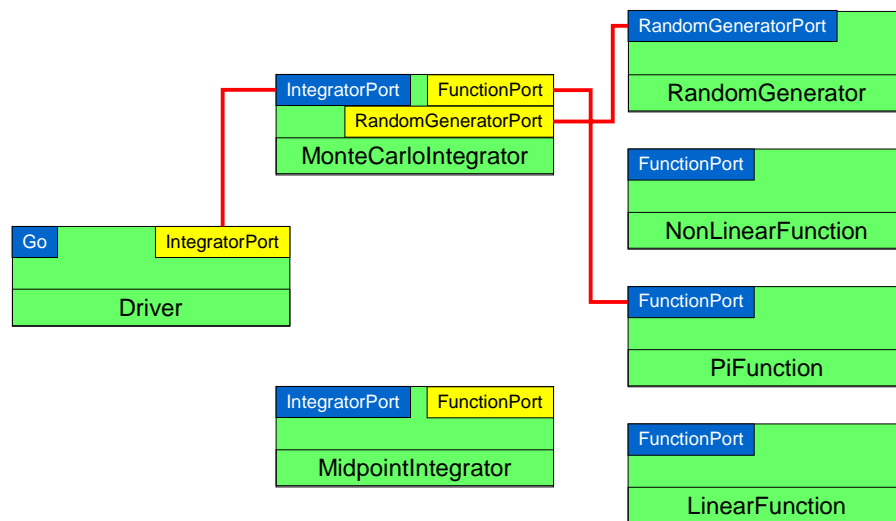
6

## Pluggability: Scenario 1



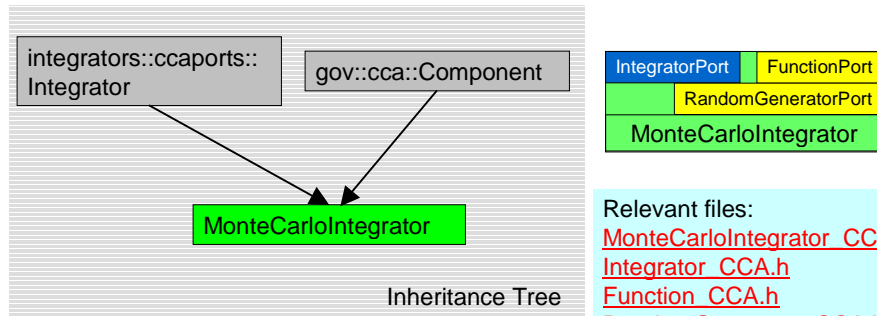
7

## Pluggability: Scenario 1



8

## MonteCarloIntegrator in Details



What makes it a component?  
 Inheritance from **gov::cca::component**

Where does **IntegratorPort** come from?  
 Inheritance from **integrators::ccaports::Integrator**

## Notes

- Inheritance from **gov::cca::Component** furnishes the only method known to the framework: **setServices()**
- **“Provides”** ports are interfaces that need to inherit from **gov::cca::Port**
- A component need to have access to definitions of methods in the ports it **“Uses”**, hence the need to include **Function\_CCA.h** and **RandomGenerator\_CCA.h**

## The Framework Role

- Framework-to-Component: **setServices()**
  - Called after the component is constructed.
  - The component's chance to identify:
    - Ports it provides – **addProvidesPort()**
    - Ports it uses – **addUsesPort()**
  - Component should not acquire the port here – Reason: it may not be there yet !!!!.
  - Also used to “shutdown” the component.

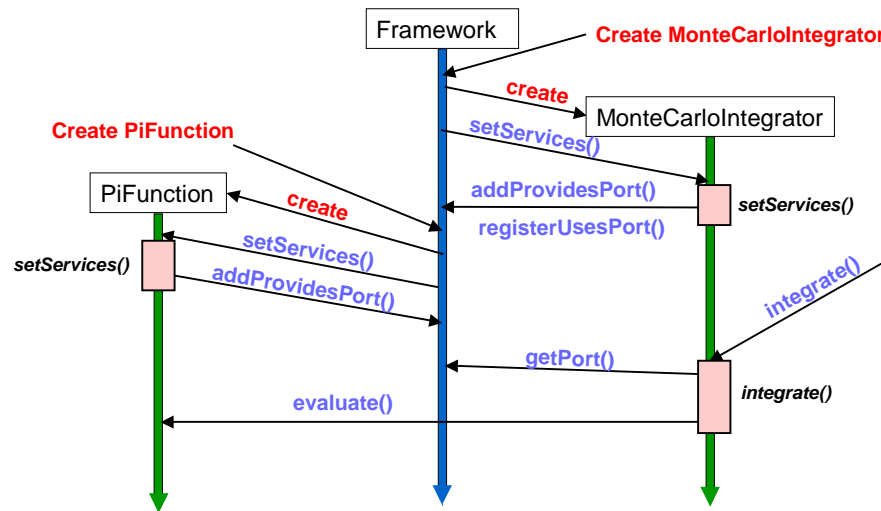
11

## Component-to-Framework

- Mainly through **Services** object passed through **setServices()**.
- **createPortInfo()**:
  - portName, portType, portProperties.
- **addProvidesPort(), registerUsesPort()**:
  - Component “pointer”, PortInfo
- **getPort()**:
  - Called by the using component.
  - Matching using portType (not name).
- **releasePort(), removeProvidesPort()**:
  - When all is done.

12

## When are calls made??



13

## Notes

- Relevant code : [MonteCarloIntegrator\\_CCA.cc](#)
- **getPort()** is called immediately before the port is used.
- The framework only knows **gov::cca::Port** , hence the **dynamic\_cast<>**.
- Making a port call: it's just a regular **virtual** call.

14



## Putting it all together

- Getting the application to do something:
  - Assembling the components into an app.
  - Launching the Application.
- App. assembly:
  - Framework need to be told what components to use, and where to find them.
  - Framework need to be told which **uses** port connects to which **provides** port.
- App execution: the **GO** port:
  - Special **provides** port used to launch the application (after connections are made).

15



## Oh Component , where art thou?.

Which components, and how to create them

```
Session Edit View Settings Help
libDriver_CCA.so
create_Driver Driver
libMonteCarloIntegrator_CCA.so
create_MonteCarloIntegrator MonteCarloIntegrator
libRandomGenerator_CCA.so
create_RandomGenerator RandomGenerator
libMidpointIntegrator_CCA.so
create_MidpointIntegrator MidpointIntegrator
libLinearFunction_CCA.so
create_LinearFunction LinearFunction
libNonlinearFunction_CCA.so
create_NonlinearFunction NonlinearFunction
libPiFunction_CCA.so
create_PiFunction PiFunction
1.1 Top
```

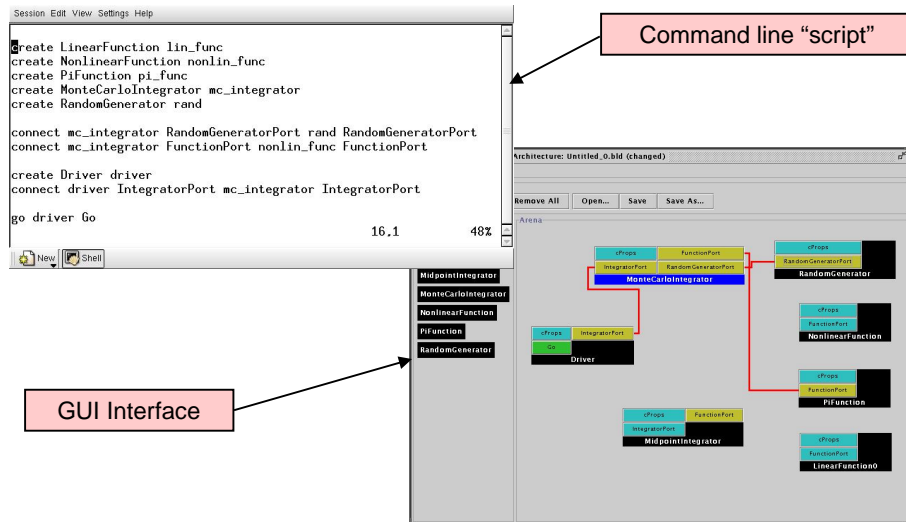
New Shell Shell No. 2

Questions?. Ask Rob

16



## App. Assembly The Ccafeine way



The screenshot displays the CCA application assembly interface. On the left, a text editor window titled "Session Edit View Settings Help" contains a script for creating and connecting components. A red box labeled "Command line 'script'" points to this window. Below the script, a "New" button and a "Shell" button are visible. On the right, a "GUI Interface" window titled "Architecture: Untitled\_0.hld (changed)" shows a component palette with various components like "MidpointIntegrator", "MonteCarloIntegrator", "NonlinearFunction", "PiFunction", and "RandomGenerator". A red box labeled "GUI Interface" points to this window. The main area of the GUI window shows a diagram of the assembled components, with red lines indicating connections between them.

```

create LinearFunction lin_func
create NonlinearFunction nonlin_func
create PiFunction pi_func
create MonteCarloIntegrator mc_integrator
create RandomGenerator rand

connect mc_integrator RandomGeneratorPort rand RandomGeneratorPort
connect mc_integrator FunctionPort nonlin_func FunctionPort

create Driver driver
connect driver IntegratorPort mc_integrator IntegratorPort

go driver Go
  
```

17

***And now, our feature presentation.***

18