Generic Component Framework

A Qt based Component Framework

Prashanth N Udupa
Email: prashanth @ vcreatelogic . com
Key Rules in Software Design

- Divide and Rule
- Establish a Constitution
- Similarity in Diversity
- Bring them together
Survey of Component Frameworks

- KDE's KParts
- Microsoft's COM
- Mozilla's XPCOM
- SOAP
- CORBA
- .... and today let me introduce another one :)

Presentation on Generic Component Framework at Akademy 2008 – Prashanth N Udupa <prashanth@vcreatelogic.com>
Introducing GCF....
What is GCF?

- A C++ framework for Qt applications. Based on Qt 4.4
- Helps compose modules of an application as a dynamically loadable entities
- Provides GUI merging capabilities
- Provides object/functionality discovery services
- Provides support for inter-process-communication.
- Provides in-built components for common tasks
- Foundation for several commercial software and the open-source VTK Designer 2
Winner of Qt Centre Programming Contest 2007
Gallery.....
Gallery....
Overview

Component

Functionality A
Functionality B
Functionality C

Interfaces to make use of the functionality

Interface for the component framework to control the component

Application

Component A
Component B
Component D
Component E

Component Manager or Component Framework
How to Divide and Rule

• Package functionality into “GCF Components”

• A GCF component offers
  – One or more widgets (subclasses of QWidget)
  – One or more objects (subclasses of QObject)
  – User interface elements (QAction, QMenu etc)
GCF's Constitution for Applications

- Have a singleton component class for each component
- Provide a GUI-XML file for each component declaring the objects, widgets, actions etc exposed by the application and their merge hints
- Serve objects, widgets, actions, menus, toolbars etc from components
- Expose component functionality via interfaces
- Have one or more objects within your object implement them
Similarity in Diversity

- **IComponent** – base class of GCF components
- **IComponentPlugin** – offers a component from a plugin
- **CComponentFactory** – dictionary of all components (instantiated, uninstantiated, active and inactive)
- Each component's GUI XML exposes one or more objects belonging to the component
- Interfaces implemented by exposed objects can be searched
The IComponent Class

• Interface Methods
  – Methods called by GCF
  – Used for GUI creation and merging
  – Used for handling activation/deactivation of components
  – Reporting component “attendance”

• Service Methods
  – Methods called by components
  – Used for object/service/component discovery
  – Used for establishing and releasing component dependency
Bring them together...

- **CComponentLoaderComponent** and **CComponentGui**
- **GUI XML File**
  - Declares objects, widgets, actions, menus, menu-strip groups
  - All declared objects are called “exposed” objects.
  - For each “exposed” object in the GUI XML file, a corresponding `create()` method is called on the IComponent.
- **Configuration capabilities**
  - Can invoke methods, set property values, make connections etc..
Demo #1
Interprocess Communication

- **Application Access Point**
  - Access to components and its objects are provided through this.

- **Remote Access Point**
  - Helps access objects in a remote application

- **Remote Object**
  - Represents a remote object
Interprocess Communication

- CAppAccessPoint
- CRemoteAppAccessPoint
- CRemoteObject
- CRemoteAppDiscovery
- CMessage
- CRemoteApp
- Based on TCP/IP
Demo #2
Getting and Installing GCF

- From www.vcreatellogic.com/oss/gcf
- SVN https://svn2.hosted-projects.com/vcreatellogic/GCF
- Supported compilers: g++, MSVC 2005, MinGW
- GCF designed, developed, maintained by VCreate Logic.
- Is distributed under GPL v2 and GPL v3
- Contributions are most definitely welcome.
- KDCF (commercial GCF) can be purchased from KDAB. For more information write to sales@kdab.net
Documentation and How to learn?

Generic Component Framework

1.6.0

Developing good user interfaces is a challenging task. Even more challenging is the task maintaining and upgrading huge UI applications. Any developer who has been involved in writing large scale UI application software would agree with this. There are a plethora of UI Widget Libraries available in the software world today to help ease out UI application development. There are also several application frameworks that help provide

**Articles**

- Basic Concepts
- Introduction to Component Frameworks
- Generic Component Framework Brief
- GUI XML Format
- Interprocess Communication

**Inbuilt Components**

- Standard Interfaces
- Help System Component
- Message Log Component
- Project Manager Component
- Property Editor Component
- Script Editor Component
- Main Window Component
- Clipboard Component
- UTILS Component
Whats next? - GCF's Roadmap....

- Support for KPart components
- Support for ActiveX components
- Support DBUS based IPC
- Support for IPC via shared-memory
- Desktop-class component deployments
- Auto-Update mechanisms
- More built-in components
Thank You

Speaker: Prashanth N Udupa

Email: prashanth @ vcreatelogic . com