ITC

Code Review

June 2, 2000

Dorota Genser

- **Repository**
  - cvsuser@d02ka.fnal.gov:/cvsroot/d0cvs
- **Packages/Versions**
  - **itc** **v00-00-13**

    Inter Task Communication package
  - **itc_event** **v00-00-02**

    Event_Message
- **Build/Platforms supported**
  - released as a part of D0 test release using KAI compiler
  - build and run on OSF1, Linux, IRIX and Windows NT
  - complied with gcc 2.95 but not fully tested
- **Dependencies**
  - KAI version 3.3f
  - ACE version 4.6
  - thread_util v00-03-02

• Code Structure

– Message

- IPC_Message is base class with internal buffer and all information needed to wrap and send data and decode it on the other side.

– Processor

- call user code when message is received
- user can register pointer or member functions for one, few or all types of messages.
- can be used with or without priority message queue and separate thread running

– Connections

- All connections (server and client) are based on one, internal class called IPC_Connection.
- user can have access to all types of connections using Connection_Handle which is a thread safe interface to IPC_Connection object.

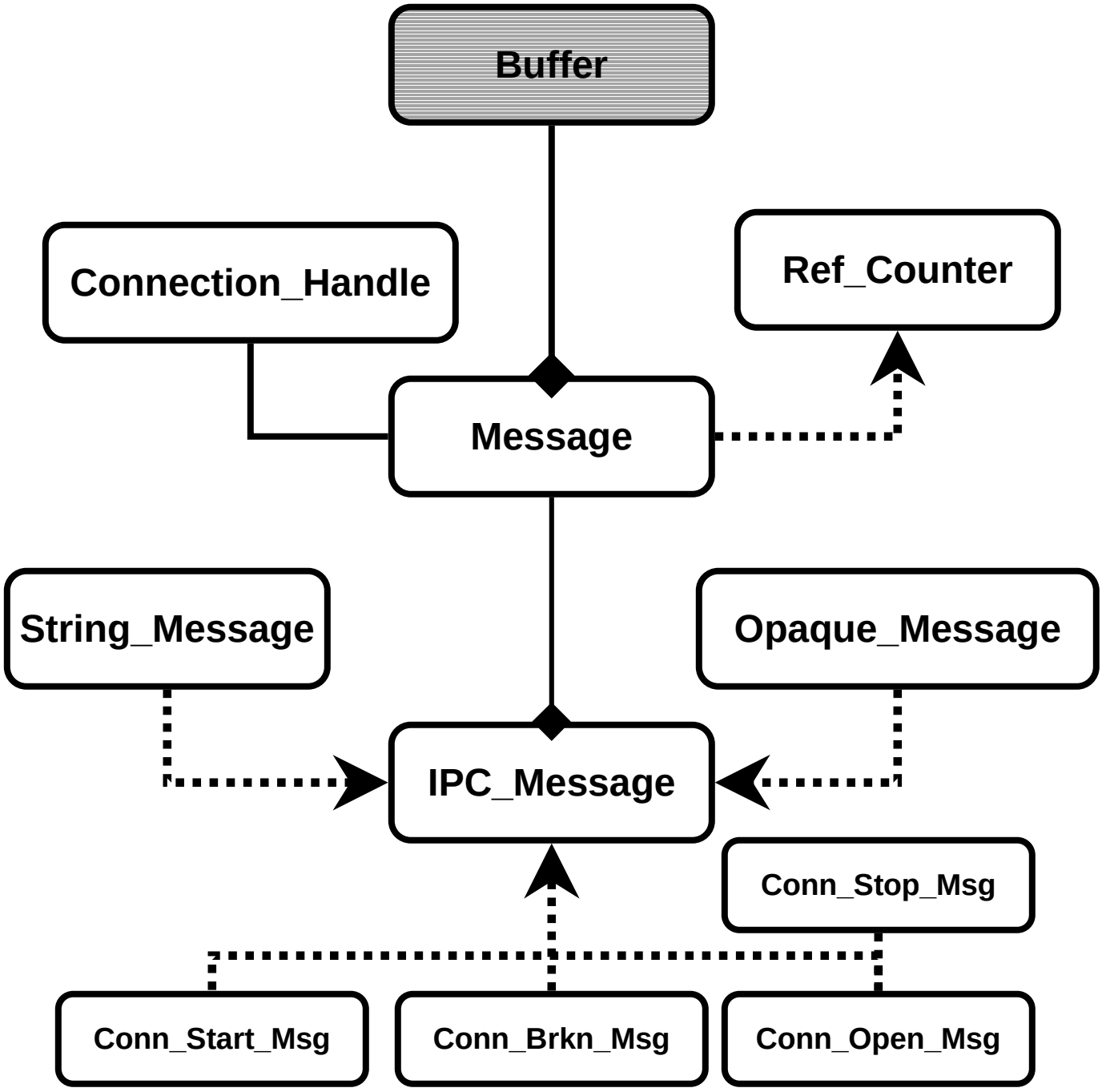
– Client

- this is small addition written in simple C++ to be used on VxWroks. It has limited functionality but it sends messages in the ITC format.

– Miscellaneous

- internal connections statistics
- debug/information macros used internally that user can use

ITC Message Classes



• ITC Message Classes

– Buffer Class

- files
  - itc/private/Buffer.hpp
  - src/Buffer.cpp
  - test/buffer_t.cpp
- manages allocated memory buffers
- main functionality:
  - creates new buffer of the given size
    - » gets block size that is bigger that requested size
    - » allocates new memory block of that block size and ads it to the list of all buffers unless such unused buffer exists already in the list
    - » takes next valid buffer ID, update buffer as used and adds it to used buffer map for fast access.
    - » returns buffer ID
  - gives access to the buffer for given ID
  - removes buffer of the given ID
    - » the buffer is marked unused and removed from used buffer map but not deleted
- additional functionality
  - check used and unused allocated memory
  - cleans some or all unused buffers to free memory

– Message and Ref_Counter Class

- files
  - itc/private/Message.hpp
  - src/Message.cpp
  - test/message_t.cpp
- Ref_Counter Class
  - simple helper class to do the reference counting and self destroy objects when the counter is zero.
- has static Buffer object
- contains message information like ID needed to access data buffer provided by Buffer class, message type, length etc.
- has Connection_Handle set for received or status messages associated with that connection

– IPC_Message Class

- user interface to internal Message class
- it holds pointer to Message object.
- IPC_Message has copy constructor and assignment operator, which increase/decrease reference count in Message object but does not delete it (Message deletes itself).

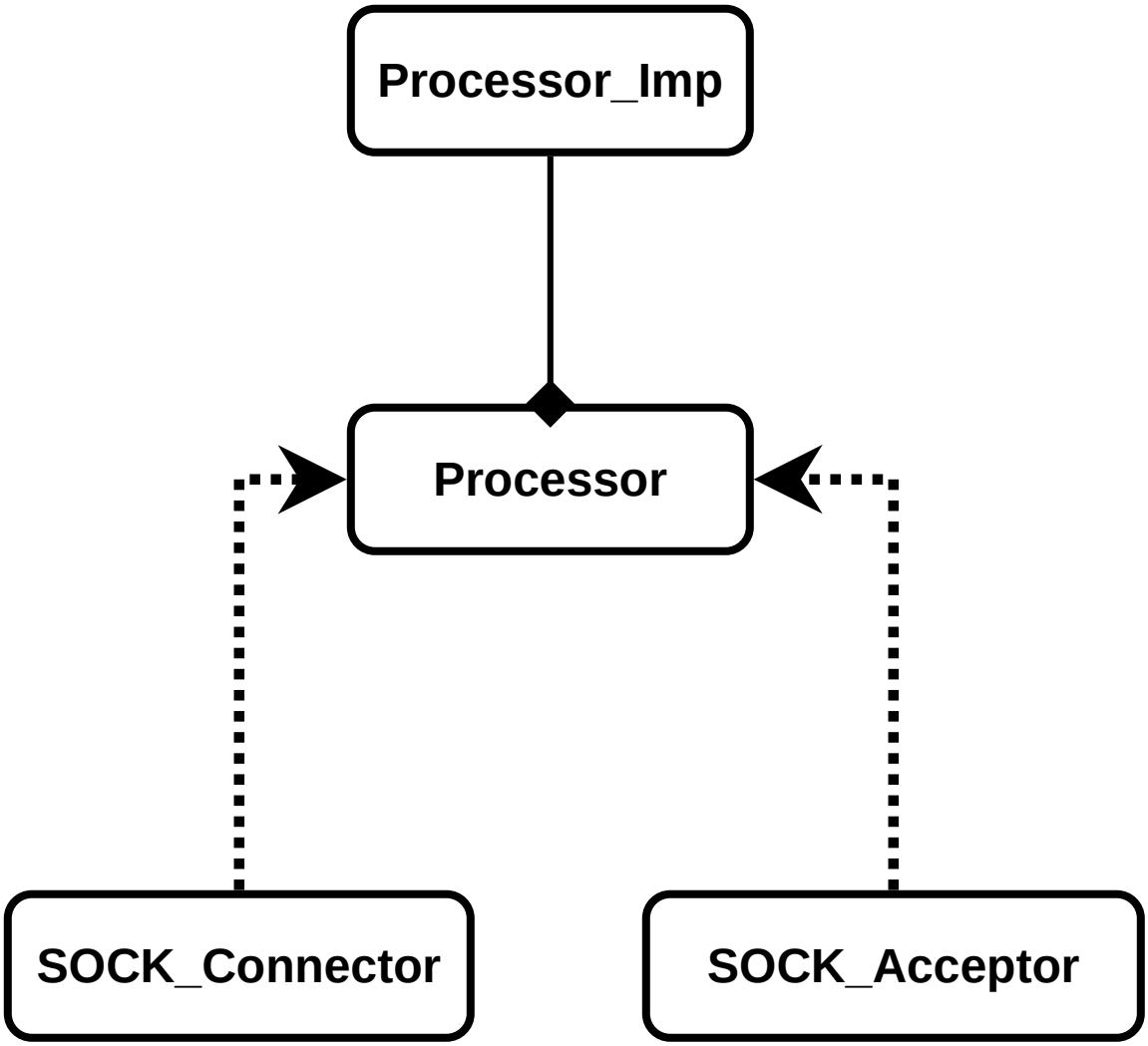
–IPC_Message Examples

- String_Message
- Opaque_Message
- Event_Message
- itc_event::Event_Message

- files
  - itc/String_Message.hpp
  - itc/Opaque_Message.hpp
  - itc/Event_Message.hpp
  - itc_event/Event_Message.hpp

  - src/Event_Message.cpp
  - itc_event/src/Event_Message.cpp

Processor



• Processor Classes

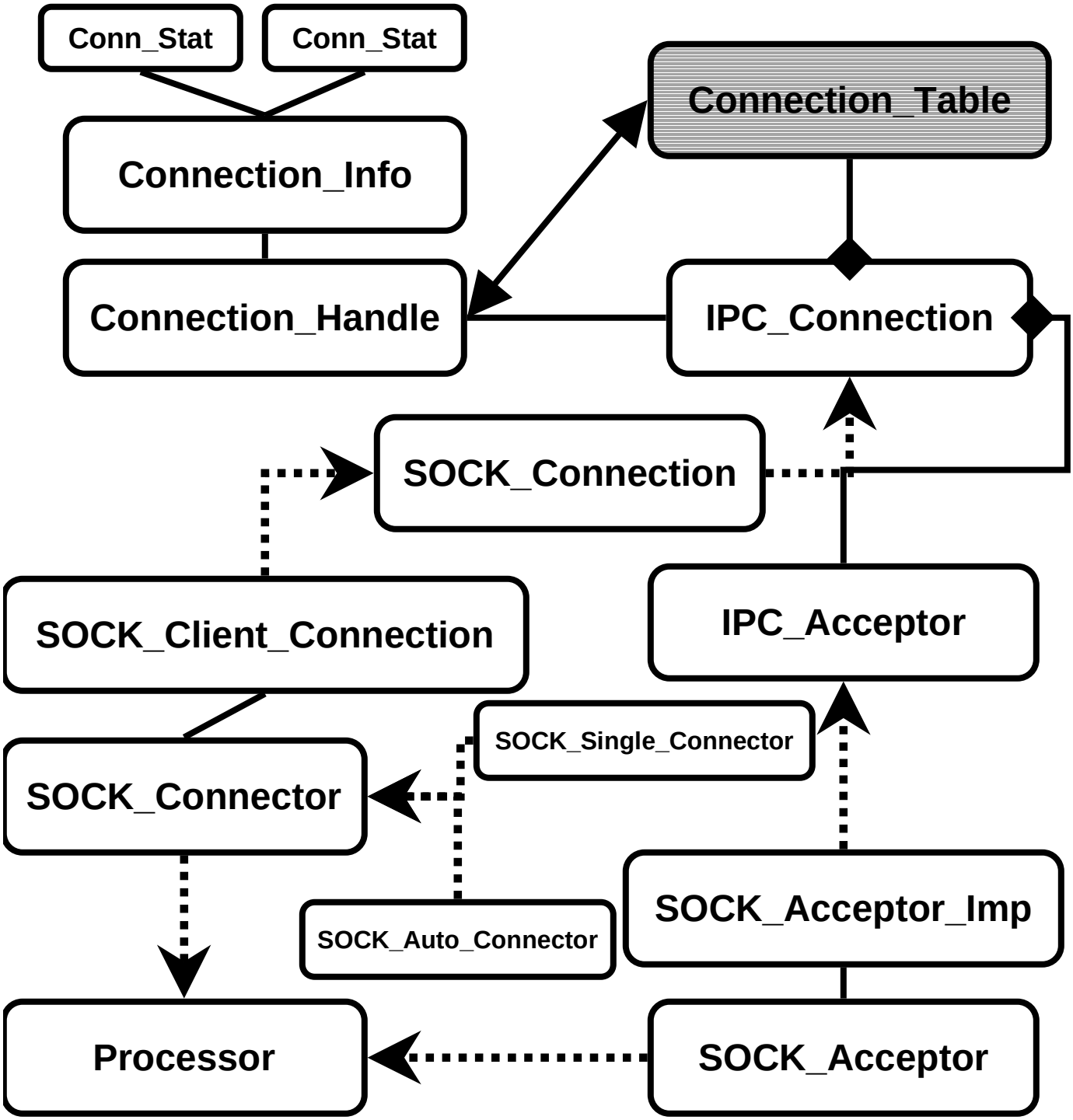
– Processor_Imp Class

- files
  - src/Processor.cpp
- internal implementation of Processor with reference counting
- allows to create Processor object (like SOCK_Connector) to be created using other Processor object. Processor_Imp object is duplicated, only the pointer is used and reference count is incremented.

– Processor Class

- files
  - itc/Processor.hpp
  - itc/Regiser_Callback.hpp
  - itc/Register_Filter.hpp
  - src/Processor.cpp
  - test/processor_t.cpp
- user interface wrapper around Processor_Imp object.

Connections



• ITC Connections Classe

– Connection_Info Class

- container class with statistics and information about other the side

– Connection_Handle Class

- user thread safe "pointer" to IPC_connection object
- has most but not all functionality of IPC_Connection class

– Connection_Table

- internal singleton class that holds map between Connection_Handle objects and pointers to IPC_Connection objects

– IPC_Connection

- abstract class that has all functionality needed to establish connection, send and receive messages etc.
- it is the most important and the most complicated class in ITC

– SOCK_Connection

- add socket specific stuff to IPC_Connection

- **SOCK_Client_Connection**
  - adds connect method to SOCK_Connection
- **SOCK_Connector**
  - user ITC base client class
- **SOCK_Single_Connector**
  - one time client connection
- **SOCK_Auto_Connector**
  - client connection with auto reconnect
- **IPC_Acceptor**
  - internal abstract server class
  - keeps and manages map of server connections
- **SOCK_Acceptor_Imp**
  - adds socket specific stuff to IPC_Acceptor
- **SOCK_Acceptor**
  - user interface to avoid ACE dependency