Dual Mode Phone

Tick
Agenda

• Legal Issues
• Dual Mode
• Skype Adapter Layer
• Qtopia Phone Server
• Qapp Arch.
• Event
• Stack Tracing
Legal Issues

• There are NO confidential materials in this slide.

• All info in this slide can be found in news or been declared in conferences.
Dual Mode Phone

- Skype
  - Skyhost
- GSM
  - Qtopia Phone Edition
Skype Adapter Layer

Skype Adapter Layer Diagram:
- MMI
- Skype Adapter
- Skype Thread
- Apiwrapper
- Socket
- Skyhost
- QCop
- Apiwrapper
- Socket
- Skyhost
Synchronies V.S. Asynchronies

• MMI has to dial many events and show animations.
• Apiwrapper is a adapter that synchronized to Skyhost.
• Queries and modifications shall not be synchronized.
• Actions shall be synchronized to Skyhost.
  - In this place we create a fake synchronize.
Qtopia Phone Server

![Diagram of Qtopia Phone Server](image_url)
Name

- PhoneServer can handle many lines at the same time.
- In PhoneServer each line has its name.
  - The name of each line follows the name of private
- PhoneServer selects specified line with name.
- PhoneLine creates private object by name.
• **PhoneLine** provides many **QueryType** that allow **Upper MMI** to query.

• The result of query will return from the SIGNAL:
  
  `queryResult(PhoneLine::QueryType type, const QString& value)`
QApp Sequence Diagram

QCoreApplication::exec

QEventLoop::ProcessEvent

QEventDispatcherUnix::ProcessEvent

QEventDispatcherUnix::doSelect

select

#read

QEventDispatcher::activateSocketNotifier

While(!d->exit)

do

While(…)

SendEvent
Moc, Signal, Slot, connect and emit

- Moc a program that implements signals, MetaTable, and functions Q_OBJECT defined.
- Signals
  - Activate a signal by name
- Slots
  - Create a table of each slot with a unique id
- Connect
  - Connect a signal and a slot to connectList
Timer

- Observer design pattern
- Each thread has its own Timer
- `startTimer` registers a Timer
QCop

- QCop was implemented by QCopChannel in which made by UNIX Socket.
- It do read and write to a socket with the scheme of QEvent.
- QCopEnvelop was sent to the socket when the QCopEnvelope instance is destructed.
- Only QApplication use it (Factory).
Stack Tracing

- `extern "C" void __cyg_profile_func_enter(void *func,void *caller) __attribute__((__no_instrument_function__));`
- `extern "C" void __cyg_profile_func_exit(void *func,void *caller) __attribute__((__no_instrument_function__));`
- `CFLAGS -finstrument-functions`
- push function entries into stack while entering a function.
- pop the function entries while leaving a function.
- show the stack log when crashed.
Q & A
~ Thanks ~
Select

- `#include <sys/select.h>`
- `int select(int n, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);`
- `FD_CLR(int fd, fd_set *set);`
- `FD_ISSET(int fd, fd_set *set);`
- Four macros are provided to manipulate the sets. `FD_ZERO` will clear a set. `FD_SET` and `FD_CLR` add or remove a given descriptor from a set. `FD_ISSET` tests to see if a descriptor is part of the set; this is useful after `select` returns.
#include <stdio.h>
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    FD_ZERO(&rfds);
    struct timeval tv;
    int retval;

    /* Watch stdin (fd 0) to see when it has input. */
    FD_ZERO(&rfds);
    FD_SET(0, &rfds);
    /* Wait up to five seconds. */
    tv.tv_sec = 5;
    tv.tv_usec = 0;

    retval = select(1, &rfds, NULL, NULL, &tv);
    /* Don't rely on the value of tv now! */
    if (retval == -1)
        perror("select()");
    else if (retval)
        printf("Data is available now.\n");
    else
        printf("No data within five seconds.\n");

    return 0;
}
Observer Pattern

```
Observer
  notify()

* (ObserverCollection)

Subject
  registerObserver(observer)
  unregisterObserver(observer)
  notifyObservers()

ConcreteObserverA
  notify()

ConcreteObserverB
  notify()
```

```
notifyObservers():
  for observer in ObserverCollection
    call observer.notify()
```
Design Pattern Discussions

- Observer → Signal Slot
- Chain of Responsibility → QCop, sendEvent
- Factory & Bridge → Decide which Dispatcher
- Template Method → Private
- Strategy → qws unix x11, glib
- Command → meta obj
- Singleton → qApp AppMap