

# Programação do ERS210A utilizando o OPEN-R SDK

**Luís Paulo Reis**

[lpreis@fe.up.pt](mailto:lpreis@fe.up.pt)

<http://www.fe.up.pt/~lpreis>

LIACC – Lab. Inteligência Artificial e Ciência de Computadores  
FEUP – Faculdade de Engenharia da Universidade do Porto

## Estrutura da Apresentação

- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

# Estrutura da Apresentação

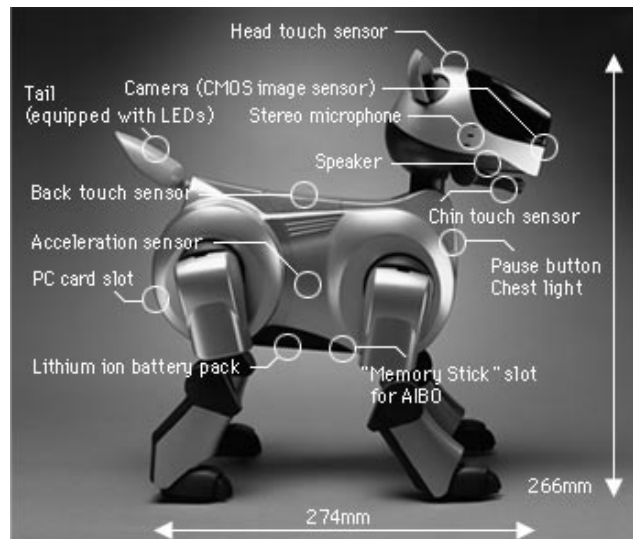
- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 3

## Plataforma Robótica AIBO -ERS210A



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 4

# OPEN-R SDK - Introdução



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 5

# OPEN-R SDK - Introdução

- OPEN-R:
  - Interface standard para robôs de entretenimento da Sony
- OPEN-R SDK (OPEN-R Software Development Kit)
  - Ambiente de desenvolvimento baseado no gcc (C++)
  - Permite construir programas que correm no ERS-210
  - Ferramentas de acesso livre
  - Arquitectura modular de Hardware e Software
  - Suporte para comunicação Wireless

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 6

# OPEN-R SDK - Introdução

- Hardware Modular:
  - Mudar a forma do Robô mudando os seus módulos (por exemplo trocar uma perna ou a cabeça!).
  - Módulos conectados por um BUS série de alta velocidade com autodeteccção da configuração do robô

# OPEN-R SDK - Introdução

- Software Modular
  - Módulo de software são “Objects”
  - Execução concorrente de objectos que comunicam uns com os outros
  - Simples substituir objectos
  - Objectos são carregados do “Memory Stick”
- Rede Wireless
  - IEEE802.11b Wireless LAN (PC Card Slot)
  - Protocolo de rede TCP/IP

## OPEN-R SDK - Introdução

- Sistema Operativo (tempo-real): Aperiodos
- Programas escritos em OPEN-R SDK
  - [www.jp.aibo.com/openr](http://www.jp.aibo.com/openr)
- Memory Sticks (gravador ou via wireless)
- OPEN-R Objects:
  - Executáveis com a extensão .BIN
  - Comunicam com outros objectos por mensagens
  - Semelhantes a processos UNIX

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 9

## OPEN-R SDK - Documentação

- Sistema:
  - Open-R SDK - Versão 1.1.3
  - Open-R SDK - Upgrade Versão 1.1.3 (release 2)
- Programas Exemplo:
  - Open-R SDK - Sample Programs (código C++)
- Documentação:
  - Open-R SDK - Installation Guide
  - Open-R SDK - Programmer's Guide
  - Open-R SDK - Model Information for ERS210
  - Open-R SDK- Level 2 - Reference Guide
  - Open-R SDK - Internet Protocol - Version 4
  - RCode Plus 251 - ERS210A

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 10

## OPEN-R SDK - Potencialidades

- Movimentar juntas do robô independentemente
- Ler os ângulos das juntas
- Ligar e desligar os leds da cabeça e cauda
- Tocar áudio PCM
- Ler a imagem da câmara
- Ler informação áudio
- Ler a data/hora, nível de bateria, etc
- Ler informação de todos os sensores do robô
- Enviar e receber dados via rede Wireless

## Conhecimentos Prévios

- Experiência em Programação em C++
- Algoritmos e Estruturas de Dados
- Inteligência Artificial e Robótica
- Utilização de ferramentas de desenvolvimento GNU
- Como usar UNIX / ferramentas Cygwin:
  - Bash, tar, ls, etc.
- Criatividade e Imaginação

## Estrutura da Apresentação

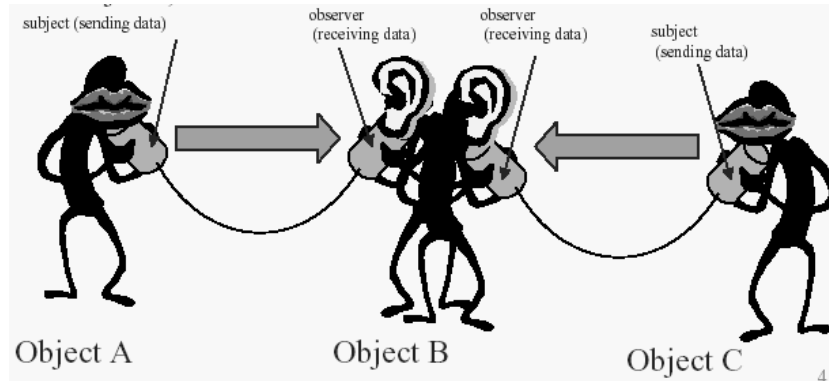
- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

## Objectos em OPEN-R

- Software OPEN-R são diversos objectos (semelhantes a ficheiros executáveis)
- Objectos correm concorrentemente
- Objectos comunicam por passagem de mensagens
- Objectos possuem múltiplos pontos de entrada

# Envio e Recepção de Dados

- Comunicação entre objectos estabelecida pelo "subject e pelo observer (incluídos no "object")



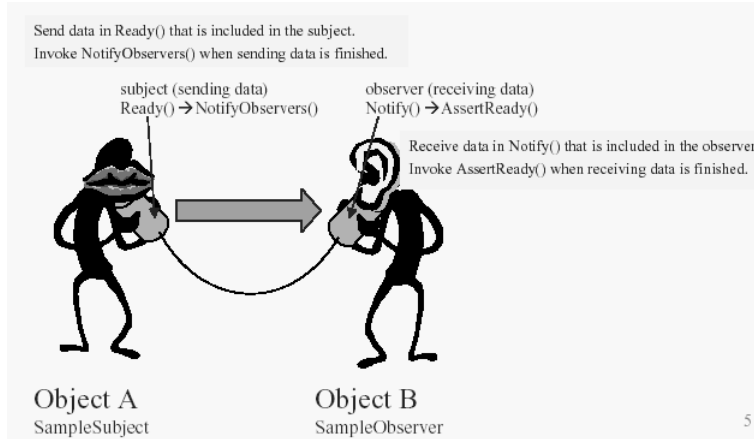
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 15

# Envio e Recepção de Dados

- Enviar dados: Ready()      Receber dados: Notify()

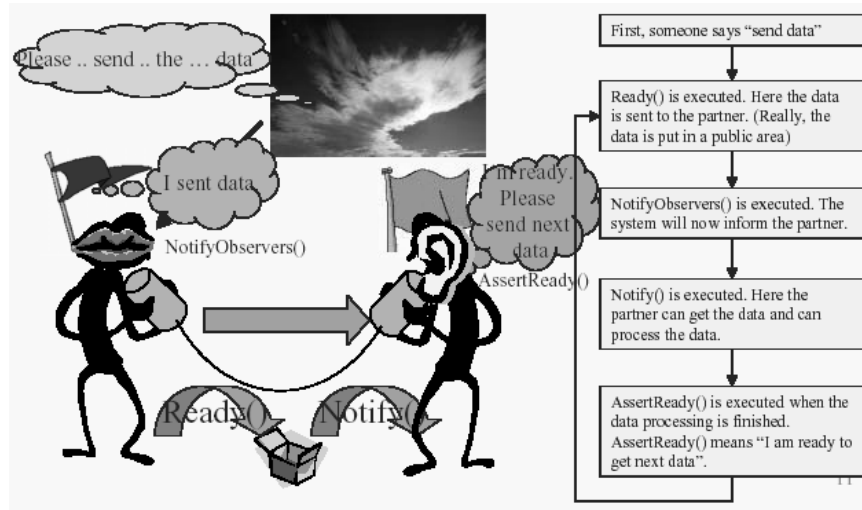


LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 16

# Envio e Recepção de Dados



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 17

# Envio e Recepção de Dados

- Para colocar os dados na área comum:
  - Subject[]-> SetData()

```
void
SampleSubject::Ready(const OReadyEvent& event)
{
    char str[32];
    strcpy(str, "!!! Hello world !!!");
    subject [sbjSendString] ->SetData(str, sizeof(str));
    subject [sbjSendString] ->NotifyObservers ();
}
```

↑  
If there are multiple subjects in one object, an array index is required.  
If there is only one subject, 'sbjSendString' should be replaced by '0'.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 18

# Envio e Recepção de Dados

- Para receber os dados da área comum:
  - Event.Data(0)

```
void
SampleObserver::Notify(const ONotifyEvent& event)
{
    const char* text = (const char *)event.Data(0);
    OSYSPRINT(("SampleObserver::Notify() %s\n", text));
    observer[event.ObsIndex()] ->AssertReady();
}

```

↑  
If there is only one observer, the array index should be '0'.

↑  
Out to console

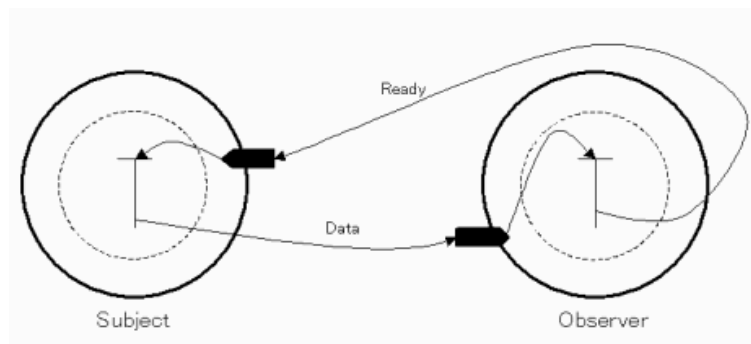
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 19

# Subjects e Observers

- Objecto que envia dados é um “Subject”
- Objecto que recebe dados é um “Observer”



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 20

## Comunicação entre Objectos

- Quando um observer está pronto envia uma mensagem `ASSERT_READY` ao subject

```
void SampleObserver::SendAssertReady () {  
    observer[obsFunc1]->AssertReady ();  
}
```

## Comunicação entre Objectos

- Quando um subject recebe um `ASSERT_READY`, pode enviar dados a um observer

```
void SampleSubject::Ready(const OReadyEvent& event) {  
    char str[32];  
    strcpy(str, "!!! Hello world !!!");  
    subject[sbjFunc2]->SetData(str, sizeof(str));  
    subject[sbjFunc2]->NotifyObservers ();  
};
```

<http://www.aibo.com/openr/>

# Comunicação entre Objectos

- Depois de um “observer” ter processado os dados, envia um novo ASSERT\_READY para receber mais dados

```
void SampleObserver::Notify(const ONotifyEvent&
                             event) {
    const char* text = (const char*) event.Data(0);
    // . . . process data . . .
    observer[obsFunc1]->AssertReady();
}
```

# Comunicação entre Objectos

```
Void SampleObserver::Notify(const ONotifyEvent& event)
{
    const char* text = (const char *)event.Data(0);
    cout << "Observer> Received: " << text << endl;
    if (strcmp(text, "shutdown") == 0) {
        OBootCondition bootCond(obcbPAUSE_SW);
        OPENR::Shutdown(bootCond);
        if (subject[sbjSendInt]->IsReady()) {
            int reply = 1;
            cout << "Observer> Sending shutdown..." << endl;
            subject[sbjSendInt]->SetData(&reply,
sizeof(reply));
            subject[sbjSendInt]->NotifyObservers();
        }
        else
            observer[event.ObsIndex()]->AssertReady();
    }
}
```



# Comunicação entre Objectos

```

Void SampleSubject::Ready(const OReadyEvent& event)
{
    char str[64];
    if (!receive_shutdown) {
        cout << "\nSubject> Enter String: ";
        cin.getline(str, 64);
        subject[sbjSendString]->SetData(str, sizeof(str));
        subject[sbjSendString]->NotifyObservers();
    }
}

Void SampleSubject::Notify(const ONotifyEvent& event)
{
    int *reply_num = (int*)event.Data(0);
    if (*reply_num == 1) {
        cout << "Subject> Received Shutdown" << endl;
        receive_shutdown = true;
    }
}

```



<http://www.aibo.com/openr/>

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 25

# Comunicação entre Objectos

To communicate between different objects, a file named stub.cfg (our 'telephone') is needed. Each object needs its own stub.cfg file.

For sending data : stub.cfg

```

ObjectName : SampleSubject
NumOfSubject : 1
NumOfObserver : 1
Service : "SampleSubject.SendString.char.S", null, Ready()
Service : "SampleSubject.DummyObserver.DoNotConnect.0", null, null

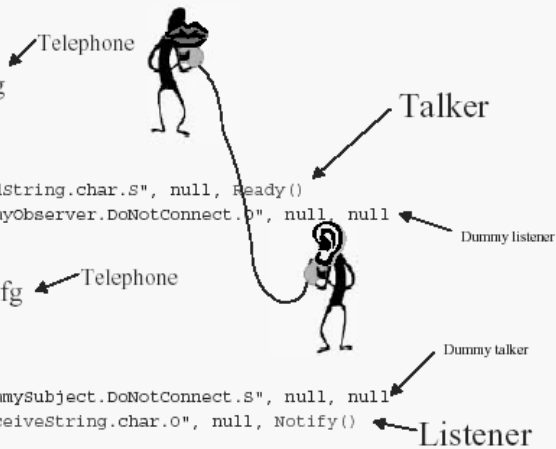
```

For receiving data : stub.cfg

```

ObjectName : SampleObserver
NumOfSubject : 1
NumOfObserver : 1
Service : "SampleObserver.DummySubject.DoNotConnect.S", null, null
Service : "SampleObserver.ReceiveString.char.0", null, Notify()

```



LIACC-FEUP

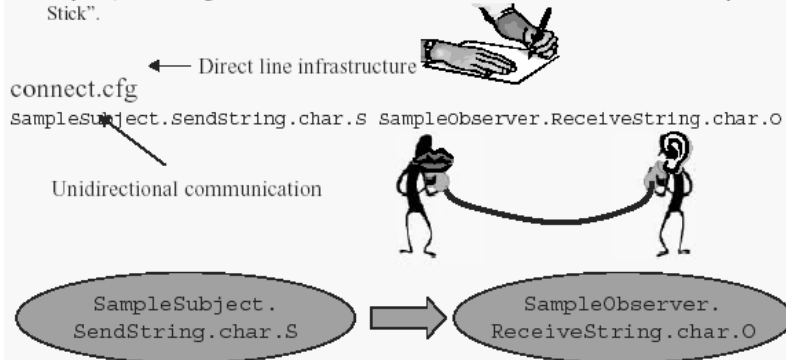
Luís Paulo Reis / 2003

Slide Nº 26

# Comunicação entre Objectos

- Linha Dedicada unidireccional entre os Objectos!

To connect objects, a file named connect.cfg is needed. Each OPEN-R application requires one (and only one) connect.cfg file. This file is located in /OPEN-R/MW/CONF on the "Memory Stick".



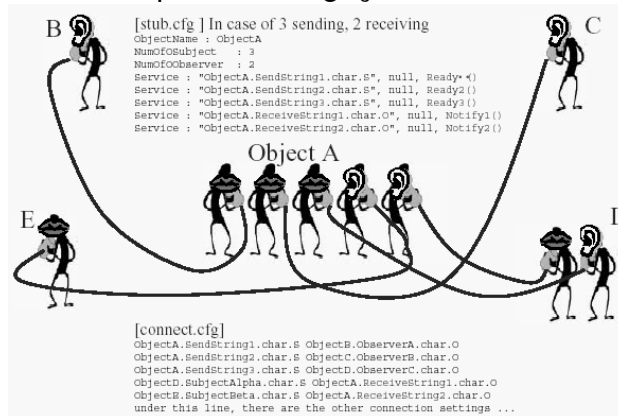
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 27

# Comunicação entre Objectos

- Um objecto pode estar conectado a múltiplos objectos (1 linha dedicada para cada ligação unidireccional!!)



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 28

# Comunicação entre Objectos

- É necessário criar uma lista com todos os objectos: `object.cfg`

The objects list is called `object.cfg`. We must create this list to inform the system which objects will exist in the application. Each OPEN-R application requires this file.

The `object.cfg` file should be placed in `/OPEN-R/MW/CONF` on the memory stick.

← The list who has telephone

`object.cfg`  
`/MS/OPEN-R/MW/OBJS/POWERMON.BIN`  
`/MS/OPEN-R/MW/OBJS/SUBJECT.BIN`  
`/MS/OPEN-R/MW/OBJS/OBSERVER.BIN`



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 29

# Exemplo de Comunicação

## “`SampleSubject.cc`”

```
#include <string.h>
#include <OPENR/OSyslog.h>
#include <OPENR/core_macro.h>
#include "SampleSubject.h"

SampleSubject::SampleSubject()
{
}

OStatus
SampleSubject::DoInit(const OSystemEvent&
    event)
{
    NEW_ALL_SUBJECT_AND_OBSERVER;
    REGISTER_ALL_ENTRY;
    SET_ALL_READY_AND_NOTIFY_ENTRY;
    return oSUCCESS;
}

OStatus
SampleSubject::DoStart(const OSystemEvent&
    event)
{
    ENABLE_ALL_SUBJECT;
    ASSERT_READY_TO_ALL_OBSERVER;
    return oSUCCESS;
}

OStatus
SampleSubject::DoStop(const OSystemEvent&
    event)
{
    DISABLE_ALL_SUBJECT;
    DRASSERT_READY_TO_ALL_OBSERVER;
    return oSUCCESS;
}

OStatus
SampleSubject::DoDestroy(const OSystemEvent&
    event)
{
    DELETE_ALL_SUBJECT_AND_OBSERVER;
    return oSUCCESS;
}

Here,
add Ready() or Notify()
as needed
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 30

# Exemplo de Comunicação

## “SampleSubject.h”

```
#include <OPENR/Object.h>
#include <OPENR/OSubject.h>
#include <OPENR/OObserver.h>
#include "def.h"

class SampleSubject : public OObject {
public:
    SampleSubject();
    virtual ~SampleSubject() {}

    OSubject*   subject[numOfSubject];
    OObserver*  observer[numOfObserver];

    virtual OStatus DoInit   (const OSystemEvent& event);
    virtual OStatus DoStart  (const OSystemEvent& event);
    virtual OStatus DoStop   (const OSystemEvent& event);
    virtual OStatus DoDestroy(const OSystemEvent& event);

    void Ready(const OReadyEvent& event);
};
```

Here, add Ready() or Notify() as needed

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 31

# Exemplo de Comunicação

## “Makefile”

```
PREFIX=/usr/local/OPEN_R_SDK
INSTALLDIR= ./MS
CXX=$(PREFIX)/bin/mipsel-linux-g++
STRIP=$(PREFIX)/bin/mipsel-linux-strip
MKBIN=$(PREFIX)/OPEN_R/bin/mkbin
STUBGEN=$(PREFIX)/OPEN_R/bin/stubgen2
MKBINFLAGS=-p $(PREFIX)
LIBS=-lObjectComm -lOPENR
CXXFLAGS=-O2 -g -I. -I$(PREFIX)/OPEN_R/include/R4000 -I$(PREFIX)/OPEN_R/include

.PHONY: all install clean

all: sampleSubject.bin

%.o: %.cc
    $(CXX) $(CXXFLAGS) -o $@ -c $^

SampleSubjectStub.cc: stub.cfg
    $(STUBGEN) stub.cfg

sampleSubject.bin: SampleSubjectStub.o SampleSubject.o sampleSubject.oef
    $(MKBIN) $(MKBINFLAGS) -o $@ $* $(LIBS)
    $(STRIP) $@

install: sampleSubject.bin
    gzip -c sampleSubject.bin > $(INSTALLDIR)/OPEN-R/MW/OBJS/SUBJECT.BIN

clean:
    rm -f *.o *.bin *.elf *.snap.cc
    rm -f SampleSubjectStub.h SampleSubjectStub.cc def.h entry.h
    rm -f $(INSTALLDIR)/OPEN-R/MW/OBJS/SUBJECT.BIN
```

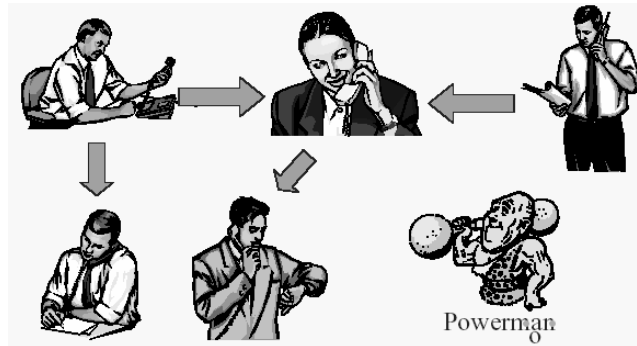
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 32

# Comunicação entre Objectos

- Numa aplicação OPEN-R existem normalmente diversos objectos, incluindo objectos que não comunicam com os outros



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 33

# Estrutura da Apresentação

- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 34

# Objectos do Sistema

- OVirtualRobotComm.Sensor
- Estrutura dos Inputs dos Sensores:
  - OSensorFrameVectorData
- OVirtualRobotComm.Effector
- Estrutura de dados para Output para os actuadores:
  - OCommandVectorData

LIACC-FEUP

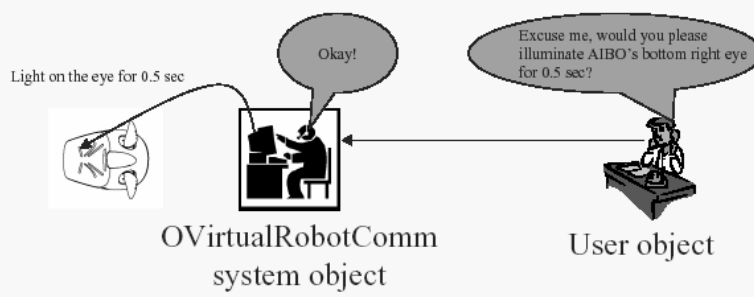
Luís Paulo Reis / 2003

Slide Nº 35

# Objectos do Sistema - Exemplo

- Exemplo: Piscar um olho do AIBO (ver sample BlinkingLed)

To make AIBO's eye blink, you must make a request to the controller of AIBO's eye (OVirtualRobotComm).



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 36

## Objectos do Sistema - Exemplo

✓ Our User Object will be called BlinkingLED (BLINKLED.BIN)

✓ The two objects we will use are POWERMON.BIN (for power issues), and BLINKLED.BIN

```
[OBJECT.CFG]
/MS/OPEN-R/MW/OBJS/POWERMON.BIN
/MS/OPEN-R/MW/OBJS/BLINKLED.BIN
```

✓ We establish an inter-object communication (direct line) between BlinkingLED.Blink and OVirtualRobotComm.Effector

```
[CONNECT.CFG]
BlinkingLED.Blink.OCommandVectorData.S OVirtualRobotComm.Effector.OCommandVectorData.O
```

✓ We configure stub.cfg (our 'telephone') for BlinkingLED it to have one sending connection (one subject).

```
[STUB.CFG]
ObjectName : BlinkingLED
NumOfOSubject : 1
NumOfOObserver : 1
Service : "BlinkingLED.Blink.OCommandVectorData.S", null, Ready()
Service : "BlinkingLED.DummyObserver.DoNotConnect.O", null, null
```

## Objectos do Sistema - Exemplo

- Forma de Comunicar com OVirtualRobotComm:
  1. Abrir cada device Led e tirar o seu "primitive ID"
  2. Criar comandos para piscar os Leds
  3. Fazer 1 e 2 na inicialização (DoInit)
  4. Preparar para apagar os Leds em DoStop

# Objectos do Sistema - Exemplo

## 1. Abrir cada device Led e tirar o seu "primitive ID"

```
static const char* const LED_LOCATOR[] = {
    "PRM:/r1/c1/c2/c3/11-LED2:11",
    "PRM:/r1/c1/c2/c3/12-LED2:12",
    "PRM:/r1/c1/c2/c3/13-LED2:13",
    "PRM:/r1/c1/c2/c3/14-LED2:14",
    "PRM:/r1/c1/c2/c3/15-LED2:15",
    "PRM:/r1/c1/c2/c3/16-LED2:16",
    "PRM:/r1/c1/c2/c3/17-LED2:17",
};
OPrimitiveID ledID[ NUM_LEDS];

void BlinkingLED::OpenPrimitives()
{
    for (int i = 0; i < NUM_LEDS; i++) {
        OStatus result = OPENR::OpenPrimitive(LED_LOCATOR[i], &ledID[i]);
    }
}
```

These are "Locators".  
They look like file names.  
Each of them are a unique reference to a certain LED.

These are "Primitive IDs"; they behave like file handles.

Loop limit is the number of LEDs (7 for ERS-210)

This function gets the primitive IDs from specified locators.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 39

# Objectos do Sistema - Exemplo

## 2. Criar comandos para piscar os Leds

```
void
BlinkingLED::NewCommandVectorData()
{
    OStatus result;
    MemoryRegionID cmdVecDataID;
    OCommandVectorData* cmdVecData;

    for (int i = 0; i < NUM_COMMAND_VECTOR; i++) {
        result = OPENR::NewCommandVectorData(NUM_LEDS,
                                              &cmdVecDataID, &cmdVecData);

        region[i] = new RCRegion(cmdVecData->vectorInfo.memRegionID,
                                cmdVecData->vectorInfo.offset,
                                (void*)cmdVecData,
                                cmdVecData->vectorInfo.totalSize);

        cmdVecData->SetNumData(NUM_LEDS);
    }
}
```

2: make two sets of commands

7: Prepare number of LEDs

Make memory regions to insert commands

7 (number of LEDs) data areas are put in one region.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 40

# Objectos do Sistema - Exemplo

## 2. Criar comandos para piscar os Leds

```

for (int j = 0; j < NUM_LEDS; j++) {
    OCommandInfo* info = cmdVecData->GetInfo(j);
    info->Set(odataLED_COMMAND2, ledID[j], 1);

    OCommandData* data = cmdVecData->GetData(j);
    OLEDCommandValue2* val = (OLEDCommandValue2*)data->value;
    if (i % 2 == 0) { // There are two command sets
        // In the 1st command set, even LEDs are ON, odd LEDs are OFF
        val[0].led = (j % 2 == 0) ? oledON : oledOFF;
    } else {
        // In the 2nd command set, even LEDs are OFF, odd LEDs are ON
        val[0].led = (j % 2 == 0) ? oledOFF : oledON;
    }
    val[0].period = 64; // 8ms * 64 = 512ms
    // illuminate the LEDs for 0.5 sec
    OSYSDEBUG(("ledID[%d] %d\n", j, val[0].led));
}
}
    
```

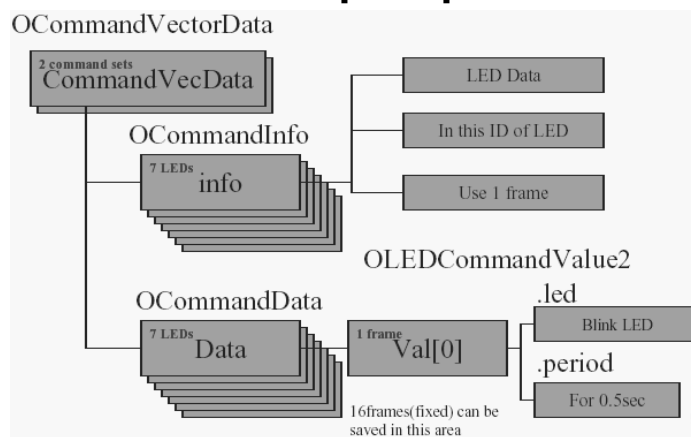
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 41

# Objectos do Sistema - Exemplo

## 2. Criar comandos para piscar os Leds



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 42

# Objectos do Sistema - Exemplo

## 3. Fazer 1 e 2 na inicialização (DoInit)

```
OStatus
BlinkingLED::DoInit(const OSystemEvent& event)
{
    OSYSDEBUG(("BlinkingLED::DoInit() %n"));

    NEW_ALL_SUBJECT_AND_OBSERVER;
    REGISTER_ALL_ENTRY;
    SET_ALL_READY_AND_NOTIFY_ENTRY;
} V.S.P. (Very special pattern)

    OpenPrimitives();
    NewCommandVectorData();

    return oSUCCESS;
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 43

# Objectos do Sistema - Exemplo

## 4. Preparar para apagar os Leds em DoStop

```
Constructor
BlinkingLED::BlinkingLED() :
    blinkingLEDState(BLS_IDLE)
{
    for (int i = 0; i < NUM_LEDS; i++) ledID[i]
        = oprimitiveID_UNDEF;
    for (int i = 0; i < NUM_COMMAND_VECTOR; i++)
        region[i] = 0;
}

OStatus
BlinkingLED::DoStart(const OSystemEvent& event)
{
    OSYSDEBUG(("BlinkingLED::DoStart() %n"));

    BlinkLED();
    blinkingLEDState = BLS_START;

    ENABLE_ALL_SUBJECT;
    ASSERT_READY_TO_ALL_OBSERVER;

    return oSUCCESS;
}

OStatus
BlinkingLED::DoStop(const OSystemEvent& event)
{
    OSYSDEBUG(("BlinkingLED::DoStop() %n"));

    blinkingLEDState = BLS_IDLE;

    DISABLE_ALL_SUBJECT;
    DEASSERT_READY_TO_ALL_OBSERVER;

    return oSUCCESS;
}

void
BlinkingLED::Ready(const OReadyEvent& event)
{
    OSYSDEBUG(("BlinkingLED::Ready() %n"));

    if (blinkingLEDState == BLS_START) {
        OSYSDEBUG(("BLS_START %n"));
        BlinkLED();
    }
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 44

# Objectos do Sistema - Exemplo

## ▪ Execução do Programa

```

void
BlinkingLED::BlinkLED()
{
    static int index = -1;

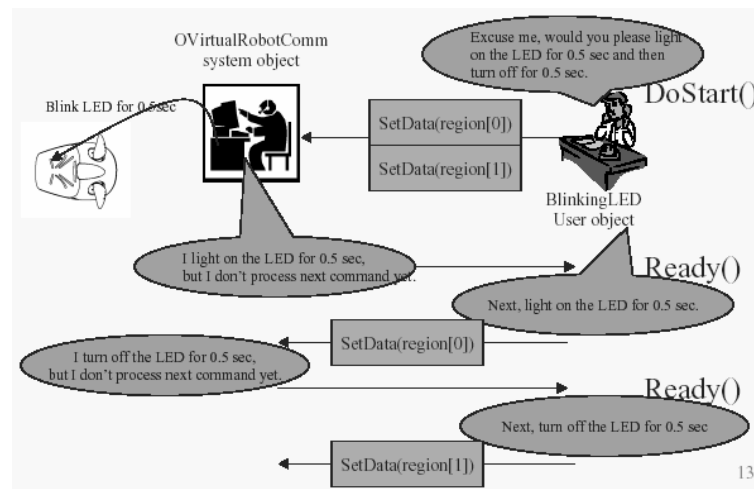
    if (index == -1) { // BlinkLED() is called first time.
        index = 0;
        subject [sbjBlink] ->SetData(region[index]);
        index++;
    }

    subject [sbjBlink] ->SetData(region[index]);
    subject [sbjBlink] ->NotifyObservers();

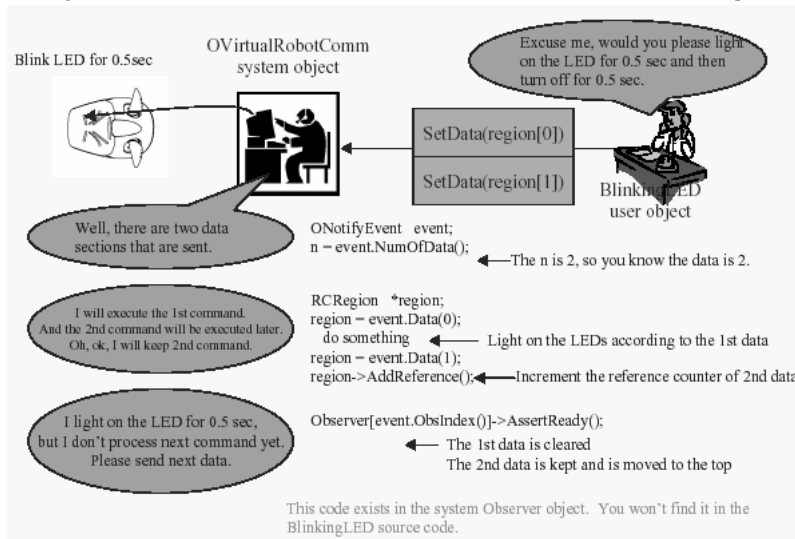
    index++;
    index = index % NUM_COMMAND_VECTOR;
    OSYSDEBUG(("index %d\n", index));
}
    
```

On the first execution, execute SetData() twice, and send two commands to blink LEDs. The second command will be "buffered" by the Observer.

# Objectos do Sistema - Exemplo



# Objectos do Sistema - Exemplo



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 47

# Objectos do Sistema - Exemplo

## Sumário:

- ✓ 1. To establish inter-object communication, create the "telephone contractor list" (OBJECT.CFG), the "direct line list" (CONNECT.CFG), and the "telephone" (stub.cfg of BlinkingLED).
- ✓ 2. To blink LEDs, make a request to the Effector observer in the OVirtualRobotComm system object.
- ✓ 3. To make a request to the Effector observer, use the OCommandVectorData data type.
- ✓ 4. To access LEDs, open 7 LEDs by using `openPrimitives()`;
- ✓ 5. Make 2 memory areas that consist of OCommandVectorData data types, where we will insert the control data.
- ✓ 6. Insert commands to blink 7 LEDs.
- ✓ 7. Set which LEDs are on or off, and how long the LEDs are lit, in the 2 areas that are created in section 5 `NewCommandVectorData()`;
- ✓ 8. Request OVirtualRobotComm object to light on or turn off the LEDs. `BlinkLED()`;

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 48

## Objectos do Sistema

- Objectos do “Layer” de sistema
- Comunica-se com o objecto  
OVirtualRobotComm.Sensor
- Para implementar isto o nosso programa é  
um Observer e OVirtualRobotComm.Sensor  
é um Subject
- Estrutura de dados para Input dos sensores  
é OSensorFrameVectorData

## OSensorFrameVectorData

- Os valores de dados (chamados “frames”)  
dos sensores são guardados aqui
- Guarda vários valores de vários sensores
- Método GetData() permite aceder a uma  
“frame” de dados

# OSensorValue

- Estrutura de dados genérica para uma “frame”
- Contido no interior de OSensorFrameData
- Deve ser “tipado” para um tipo apropriado
- Exemplo: Para valores das juntas deve ser “tipado” par OJointValue
- Exemplo: OForce para valores de pressão

```
struct OForce {  
    slongword value;           // units in 10-6 N  
    word signal;  
    word padding[5];  
};
```

# Sensores - Utilização

- Obter dados dos sensores é idêntico a comunicar com outros objectos:
  1. Seleccionar Sensores a Ler
  2. Enviar ASSERT\_READY a OVirtualRobotComm.Sensor
  3. Obter OSensorFrameVectorData de OVirtualRobotComm.Sensor
  4. Utilizar os métodos em OSensorFrameVectorData para obter os OSensorValue
  5. “Tipar” os OSensorValue para o tipo apropriado a cada sensor

# Movimentação das Juntas

- Objecto que recebe os comando chama-se OVirtualRobotComm.Effector
- Tipo de dados é OCommandVectorData
  - Pode conter múltiplos comandos

```
struct OCommandVectorData {
    ODataVectorInfo vectorInfo;
    OCommandInfo info[1];
    void SetNumData(size_t ndata){ vectorInfo.numData = ndata; }
    OCommandInfo* GetInfo(int index) { return &info[index]; }
    OCommandData* GetData(int index) {
        return (OCommandData*)((byte*)&vectorInfo +
            info[index].dataOffset);
    }
};
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 53

# OCommandInfo

- Contém a informação de um comando específico

```
struct OCommandInfo {
    ODataType type;
    OPrimitiveID primitiveID;
    longword frameNumber;
    size_t numFrames;
    size_t frameSize;
    size_t dataOffset;
    size_t dataSize;
    longword padding[1];
    void Set(ODataType t, OPrimitiveID id, size_t nframes)
    {
        type = t; primitiveID = id;
        numFrames = nframes;
    }
};
```

<http://www.aibo.com/openr/>

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 54

## Estrutura da Apresentação

- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

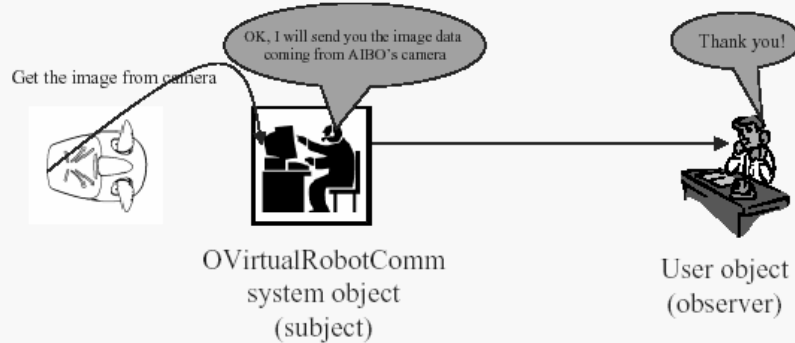
## Visão do AIBO – Exemplo 1

- Analisar imagem da câmara do AIBO
- O Sample “Image Observer” – Tipos de Imagem
- Comunicação entre objectos com imagem
- Retirar a Imagem de OVirtualRobotComm
  - Abrir o Sensor de Imagem (Câmara) e retirar o primitive ID (em DoInit)
  - Construir uma tabela de detecção de cor (em DoInit)
  - Parar a recepção de imagem (em DoStop)
- Receber dados em event.Data(0) e fazer AssertReady() para receber os próximos dados

# Visão do AIBO – Exemplo 1

## Retirar Dados da Câmara

To retrieve the camera data, you must make a request to the controller of AIBO's camera (OVirtualRobotComm).



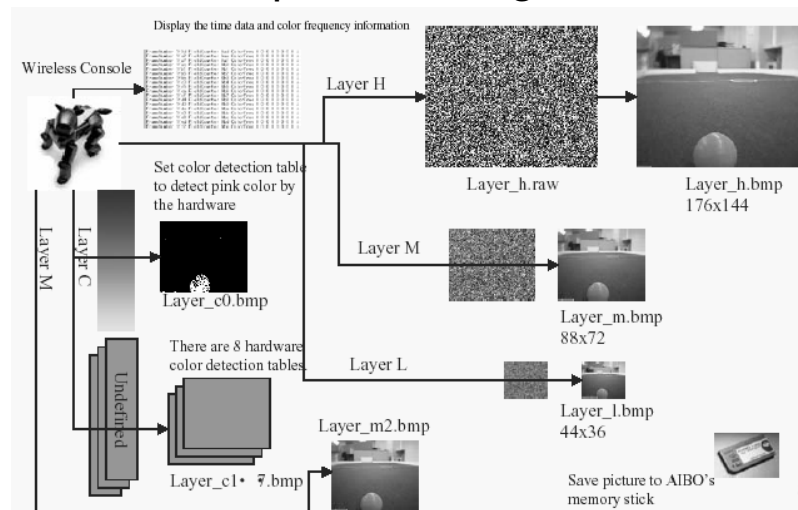
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 57

# Visão do AIBO – Exemplo 1

## Tipos de Imagem



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 58

## Visão do AIBO – Exemplo 1 Comunicação entre Objectos

- ✓ Our user object will be called ImageObserver (IMAGEOBS.BIN)
- ✓ The two objects we will use are POWERMON.BIN (for power issues), and IMAGEOBS.BIN

```
[OBJECT.CFG]
/MS/OPEN-R/MW/OBJS/POWERMON.BIN
/MS/OPEN-R/MW/OBJS/IMAGEOBS.BIN
```

- ✓ We establish an inter-object communication (direct line) between ImageObserver.Image and OVirtualRobotComm.FbkImageSensor

```
[CONNECT.CPG]
OVirtualRobotComm.FbkImageSensor.OFbkImageVectorData.S ImageObserver.Image.OFbkImageVectorData.O
```

- ✓ We configure stub.cfg (our 'telephone') for ImageObserver to have one receiving connection (one observer).

```
[STUB.CFG]
ObjectName : ImageObserver
NumOfSubject : 1
NumOfObserver : 1
Service : "ImageObserver.Image.OFbkImageVectorData.O", null, Notify()
Service : "ImageObserver.DummySubject.DoNotConnect.S", null, null
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 59

## Visão do AIBO – Exemplo 1 Estrutura Geral da Captura de Imagem

1. Open image sensor and get its primitive ID.
2. Set color detection table to detect the color pink.
3. #1 and #2 are established in the initialization section (DoInit).
4. Prepare a flag value to stop getting image data when Dostop() is invoked.
5. Oops! Do we need a memory area to save the image? The subject gets the area for saving the image data.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 60

## Visão do AIBO – Exemplo 1

### Abrir o Sensor de Imagem e retirar “Primitive ID”

```

static const char* const FBK_LOCATOR
    = "PRM:/r1/c1/c2/c3/i1-FbkImageSensor:F1";
OPrimitiveID      fbkID;
    
```

} This is a “Locator”.  
It looks like a file name.  
It references AIBO’s camera.

```

void
ImageObserver::OpenPrimitive()
{
    OStatus result = OPENR::OpenPrimitive(FBK_LOCATOR, &fbkID);
}
    
```

Get primitive ID from the locator:

↗

This is a “Primitive ID”; they behave like file handles

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 61

## Visão do AIBO – Exemplo 1

### Tabela de Detecção (“Pink”)

```

void
ImageObserver::SetCdtVectorDataOfPinkBall()
{
    OStatus result;
    MemoryRegionID cdtVecID;
    OCdtVectorData* cdtVec;
    OCdtInfo*      cdt;

    result = OPENR::NewCdtVectorData(&cdtVecID, &cdtVec);
    //Prepare cdt vector data
    cdtVec->SetNumData(1); // Use only 1 channel (8 channels can be used)
    cdt = cdtVec->GetInfo(0); // Use the 1st channel      8 colors can be detected
    cdt->Init(fbkID, ocdtCHANNEL0); //Initialize          at the same time by hardware

    // cdt->Set(Y_segment, Cr_max, Cr_min, Cb_max, Cb_min)
    cdt->Set( 0, 230, 150, 190, 120);
    cdt->Set( 1, 230, 150, 190, 120);
    cdt->Set( 2, 230, 150, 190, 120);
    // Omission...
}
    
```

} This is the setting for detecting the color pink.  
(Details will be described later)

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 62

## Visão do AIBO – Exemplo 1

### Tabela de Detecção (“Pink”)

```
// Omission...
cdt->Set(19, 230, 150, 190, 120);
cdt->Set(20, 230, 160, 190, 120);
// Omission...
cdt->Set(30, 230, 160, 190, 120);
cdt->Set(31, 230, 160, 190, 120);

result = OPENR::SetCdtVectorData(cdtVecID);
//Set color information to color detection hardware by using cdt vector

result = OPENR::DeleteCdtVectorData(cdtVecID);
//Now, cdt vector isn't needed, so it is deleted.
}
```

This is the setting for detecting the color pink.  
(Details will be described later)

## Visão do AIBO – Exemplo 1

### Construção da Secção Dolnit

- Chamada das funções em Dolnit

```
OStatus
ImageObserver::DoInit(const OSystemEvent& event)
{
    NEW_ALL_SUBJECT_AND_OBSERVER;
    REGISTER_ALL_BNTRY;
    SET_ALL_READY_AND_NOTIFY_ENTRY;

    OpenPrimitive();
    SetCdtVectorDataOfPinkBall();

    return OSUCCESS;
}
```

This is V.S.P.  
(Very special pattern)

## Visão do AIBO – Exemplo 1 Construção da Secção DoStop

```
constructor
ImageObserver::ImageObserver() :
    imageObserverState (IOS_IDLE),
    fbkID (oprimitiveID_UNDEF)
{
}

OStatus
ImageObserver::DoStart(const OSystemEvent&
    event)
{
    imageObserverState = IOS_START;

    ENABLE_ALL_SUBJECT;
    ASSERT_READY_TO_ALL_OBSERVER;

    return oSUCCESS;
}

OStatus
ImageObserver::DoStop(const OSystemEvent&
    event)
{
    imageObserverState = IOS_IDLE;

    DISABLE_ALL_SUBJECT;
    DEASSERT_READY_TO_ALL_OBSERVER;

    return oSUCCESS;
}

void
ImageObserver::Notify(const ONotifyEvent&
    event)
{
    if (imageObserverState == IOS_IDLE) {
        return; // do nothing
    }
    OFbkImageVectorData* fbkImageVectorData
        = (OFbkImageVectorData*)event.Data(0);
    // Do many (will explain later)
    observer[event.ObsIndex()] ->AssertReady
        (event.SenderID());
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 65

## Visão do AIBO – Exemplo 1 Recepção dos Dados

- Receber Dados: event.Data[0]
- Requisitar novos Dados: AssertReady

```
void
ImageObserver::Notify(const ONotifyEvent& event)
{
    if (imageObserverState == IOS_IDLE) {
        return; // do nothing
    }
    OFbkImageVectorData* fbkImageVectorData
        = (OFbkImageVectorData*)event.Data(0);
    // process the image data, here
    // Save data to memory stick
    observer[event.ObsIndex()] ->AssertReady(event.SenderID());
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 66

# Visão do AIBO – Exemplo 1

## Sumário

- ✓ 1. To establish inter-object communication, create the “telephone contractor list” (OBJECT.CFG), the “direct line list” (CONNECT.CFG), and the “telephone” (stub.cfg of ImageObserver).
- ✓ 2. The image can be received from the FbkImageSensor subject in the OVirtualRobotComm object.
- ✓ 3. Open the image sensor to get the image.  
`OpenPrimitives()` ;
- ✓ 4. Set the color detection table to detect the color pink before receiving data.  
`SetCdtVectorDataOfPinkBall()` ;
- ✓ 5. Receive image data via `event.Data(0)` in `Notify()` . The data type is `OFbkImageVectorData`.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 67

# Visão do AIBO

## RGB vs YCbCr

- ✓ AIBO’s camera expresses colors through Y(brightness), Cr(difference from red), Cb(difference from blue) → YCbCr format
- ✓ You can convert to the widely understood RGB format by following this numerical formula (The range of Y is 0.0~1.0, the range of Cb, Cr is -1.0~1.0)  
 $R = 255.0 * (Y + Cr)$ ;  
 $G = 255.0 * (Y - 0.51 * Cr - 0.19 * Cb)$ ;  
 $B = 255.0 * (Y + Cb)$ ;  
The R,G,B values should then be constrained (“clipped”) to the range 0~255.
- ✓ AIBO’s camera sends 4 types of image data (layer)

LayerH	High resolution image data YCbCr 176x144
LayerM	Middle resolution image data YCbCr 88x72
LayerL	Low resolution image data YCbCr 44x36
LayerC	Result from color detection engine (CDT image) 88x72
- ✓ AIBO’s color detection engine is able to distinguish 8 colors (8 channels) at the same time

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 68

# Visão do AIBO

## A Tabela de Detecção de Cores

✓ Channel and color detection table

There are 8 channels.

```
// cdt->Set(Y_segment, Cr_max, Cr_min, Cb_max, Cb_min)
```

0			
1			
2			
3			
4			
5			
6			
7			

0		Range of Cr	Range of Cb
1		Range of Cr	Range of Cb
2	Y-16-23	Range of Cr	Range of Cb
3	Y-24-31	Range of Cr	Range of Cb
4	Y-32-39	Range of Cr	Range of Cb
5	Y-40-47	Range of Cr	Range of Cb
6	Y-48-55	Range of Cr	Range of Cb
7	Y-56-63	Range of Cb	Range of Cb

```
cdt->Set(19, 230, 150, 190, 120);
cdt->Set(20, 230, 160, 190, 120);
```

30	Y-240-247	Range of Cb	Range of Cb
31	Y-248-255	Range of Cb	Range of Cb

The brightness index table has 32 stages.

LIACC-FEUP

Luis Paulo Reis / 2003

Slide Nº 69

# Visão do AIBO – Exemplo 2

## Gravação de Imagens em Bitmap

```
void
ImageObserver::Notify(const CNotifyEvent& event)
{
    static int counter = 0;

    if (ImageObserverState == IOS_IDLE) { } Terminate image retrieval when DoStop() is invoked
        return; // do nothing
    }

    OFbkImageVectorData* fbkImageVectorData
        = (OFbkImageVectorData*)event.Data(0); // Here, receive image data

    BMP bmp;
    if (counter == 0) { // The 1st round

        OSYSPRINT(("SAVE LAYER_H.BMP ... "));
        bmp.SaveYCrCb2RGB("/MS/OPEN-R/MW/DATA/P/LAYER_H.BMP",
            fbkImageVectorData, ofbkImageLAYER_H); ← Save LayerH data to memory stick with conversion to BMP format
        OSYSPRINT(("DONE\n"));

        OSYSPRINT(("SAVE LAYER_H.RAW ... "));
        SaveRawData("/MS/OPEN-R/MW/DATA/P/LAYER_H.RAW",
            fbkImageVectorData, ofbkImageLAYER_H); ← Save LayerH data to memory stick without any conversion
        OSYSPRINT(("DONE\n"));
    }
}
```

LIACC-FEUP

Luis Paulo Reis / 2003

Slide Nº 70

## Visão do AIBO – Exemplo 2 Gravação de Imagens em Bitmap

```

} else if (counter == 1) { // The 2nd round
  OSYSPRINT("SAVE LAYER_M.BMP ... ");
  bmp.SaveYCrCb2RGB("/MS/OPEN-R/MW/DATA/P/LAYER_M.BMP",
    fbkImageVectorData, ofbkImageLAYER_M);
  OSYSPRINT("DONE\n");
} else if (counter == 2) { // The 3rd round
  OSYSPRINT("SAVE LAYER_L.BMP ... ");
  bmp.SaveYCrCb2RGB("/MS/OPEN-R/MW/DATA/P/LAYER_L.BMP",
    fbkImageVectorData, ofbkImageLAYER_L);
} else if (counter == 3) { // The 4th round
  OSYSPRINT("SAVE LAYER_C ... ");
  bmp.SaveLayerC("/MS/OPEN-R/MW/DATA/P/LAYER_C", fbkImageVectorData);
  OSYSPRINT("DONE\n");
  OSYSPRINT("SAVE LAYER_M2.BMP ... ");
  bmp.SaveYCrCb2RGB("/MS/OPEN-R/MW/DATA/P/LAYER_M2.BMP",
    fbkImageVectorData, ofbkImageLAYER_M);
  OSYSPRINT("DONE\n");
} else { // More than 5th round
  PrintTagInfo(fbkImageVectorData);
}
counter++;
observer[event.ObsIndex()->AssertReady(event.SenderID()); // I'm ready, send next data!
}

```

← Save LayerM data to memory stick with conversion to BMP format  
 ← Save LayerL data to memory stick with conversion to BMP format  
 ← Save LayerC data to memory stick using 8 separate files (black & white)

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 71

## Visão do AIBO – Exemplo 2 Conversão de YCbCr para RGB

```

bool
BMP::SaveYCrCb2RGB(char* path,
  OpbkImageVectorData* imageVec,
  OpbkImageLayer layer)
{
  if (layer == ofbkImageLAYER_C)
    return false;

  byte pixels[3];
  OpbkImageInfo* info
  = imageVec->GetInfo(layer);
  byte* data
  = imageVec->GetData(layer);

  OpbkImage yImage(info, data,
    ofbkImageBAND_Y); //Get (Y) Brightness

  OpbkImage crImage(info, data,
    ofbkImageBAND_Cr); //Get Cr

  OpbkImage cbImage(info, data,
    ofbkImageBAND_Cb); //Get Cb

  BMPHeader header;
  BMPInfoHeader infoheader;

  infoheader.width = yImage.Width();
  infoheader.height = yImage.Height();
  infoheader.imagesize
  = yImage.Width() * yImage.Height() * 3;
  header.size = infoheader.imagesize + 54;

  FILE* fp = fopen(path, "w"); //Saving data to MS
  if (fp == 0) return false;

  SaveBMPHeader(fp, header);
  SaveBMPInfoHeader(fp, infoheader);

  for (int y = infoheader.height - 1; y >= 0; y--)
  { //Up side down Note height -1 (explain later)
    for (int x = 0; x < infoheader.width; x++)
    {
      YCrCb2RGB(yImage.Pixel(x, y),
        crImage.Pixel(x, y),
        cbImage.Pixel(x, y),
        &pixels[0_PIXEL], &pixels[0_PIXEL], &pixels[0_PIXEL]);
      // Here, convert YCbCr to RGB
      fwrite(pixels, 1, 3, fp);
    }
  }

  fclose(fp);
  return true;
}

```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 72

## Visão do AIBO – Exemplo 2

### Utilização dos Dados da Detecção – Layer C

```

bool
EMP::SaveLayerC(char* basepath,
  OFbkImageVectorData* imageVec)
{
  byte pixels[3];
  char path[128];
  OFbkImageInfo* info
  = imageVec->GetInfo(ofbkimageLAYER_C);
  byte* data
  = imageVec->GetData(ofbkimageLAYER_C);

  OFbkImage cdtImage(info, data,
    ofbkimageRAND_CDT);

  BMPHeader header;
  BMPInfoHeader infoheader;

  infoheader.width = cdtImage.Width();
  infoheader.height = cdtImage.Height();
  infoheader.imagesize = cdtImage.Width()
  * cdtImage.Height() * 3;

  header.size = infoheader.imagesize + 54;

  for (int i = 0; i < ocdtNUM_CHANNELS; i++) {
    // execute 8 times (separate each 8Bit)
    byte plane = 0x01 << i; // Create mask data
    sprintf(path, "%s%d.BMP", basepath, i);
    // create file names (8)
    FILE* fp = fopen(path, "w");
    if (fp == 0) return false;

    SaveBMPHeader(fp, header);
    SaveBMPInfoHeader(fp, infoheader);

    for (int y = infoheader.height - 1; y >= 0; y--)
    { //up side down , Note hight -1
      for (int x = 0; x < infoheader.width;
        x++) {
        if (cdtImage.Pixel(x, y) & plane) {
          pixels[R_PIXEL] = pixels[G_PIXEL]- pixels[B_PIXEL] - 255;
        } else { //white
          pixels[R_PIXEL] = pixels[G_PIXEL]- pixels[B_PIXEL] = 0;
        } //black
        fwrite(pixels, 1, 3, fp);
      }
    }
    fclose(fp);
  }
  return true;
}

```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 73

## Visão do AIBO – Exemplo 2

### Introdução do Tempo, Data e Frequên. Cores

The last line of the image data is NOT picture data, it include field counter (timer) and color frequency information.

```

void
ImageObserver::PrintTagInfo(OFbkImageVectorData* imageVec)
{
  OFbkImageInfo* info = imageVec->GetInfo(ofbkimageLAYER_H);
  byte* data = imageVec->GetData(ofbkimageLAYER_H);
  OFbkImage yImage(info, data, ofbkimageRAND_Y);
  OSYSPRINT("FrameNumber %X FieldCounter %X ColorFreq ",
    imageVec->GetInfo(0)->frameNumber, yImage.FieldCounter());
  // frameNumber is time frame (8m sec unit) from starting AIBO, the max value is 0x0ffffff0;
  // 24days (AIBO will shutdown before 24days due to low battery)
  // FieldCounter is a counter for updating camera data (the camera data is updated each 40m sec
  // units) the max value is 0xffff (43minutes), the counter will be 0x0000 when it goes beyond
  // the max value.
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL0));
  // ColorFrequency means how many pixels are detected with the specific color. This value is
  // calculated by (detected pixels / 16). If it is more than 4080 pixels then the value
  // is 255
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL1));
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL2));
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL3));
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL4));
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL5));
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL6));
  OSYSPRINT("%d ", yImage.ColorFrequency(ocdtCHANNEL7));
  OSYSPRINT("\n");
}

```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 74

## Visão do AIBO – Exemplo 2

### Gravação de Imagens Raw

```
void
ImageObserver::SaveRawData(char* path,
                           OFbkImageVectorData* imageVec, OFbkImageLayer layer)
{
    OFbkImageInfo* info = imageVec->GetInfo(layer);
    byte* data = imageVec->GetData(layer);

    size_t size;
    if (layer == ofbkimageLAYER_C) {
        size = info->width * info->height; // If LayerC then width x height
    } else {
        size = 3 * info->width * info->height;
        // If the other then width x height x 3(Y,Cr,Cb)
    }

    FILE* fp = fopen(path, "w");
    if (fp == 0) return;
    fwrite(data, 1, size, fp); // write all at once
    fclose(fp);
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 75

## Visão do AIBO – Exemplo 2

### Sumário

- ✓ 1. To establish inter-object communication, create the “telephone contractor list” (OBJECT.CFG), the “direct line list” (CONNECT.CFG), and the “telephone” (stub.cfg of ImageObserver).
- ✓ 2. The image can be received from the FbkImageSensor subject in the OVirtualRobotComm object.
- ✓ 3. Open the image sensor to get the image.  
openPrimitives();
- ✓ 4. Set color detector table to detect the color pink before receiving data. 8 colors can be detected at the same time.  
setCdtVectorDataOfPinkBall();
- ✓ 5. To set the color detection table, the color format is YCbCr. The brightness table has 32 stages; set the range of Cr, Cb into each index.  
cdt->Set(Y\_segment, cr\_max, cr\_min, cb\_max, cb\_min);
- ✓ 6. The images that can be received are in YCbCr format and are available in three sizes (High, Middle, and Low resolution). To convert to BMP format:  
a. convert to RGB format   b. flip upside down   c. delete last line
- ✓ 7. If the color detection table is set, the target color information can be viewed in LayerC by bitmask (0/1) information. LayerC is an 8bit plane.
- ✓ 8. The last line of the image is the time data and color frequency information (8 channels).

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 76

# Estrutura da Apresentação

- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

## Combinação de Objectos Programa de Captura de Imagem (PCI)

- ✓ The main feature of OPEN-R provides modularized software
- ✓ We will make a sample program called "ImageCapture"
- ✓ To make the ImageCapture program, you only need to gather 4 independent functions together.
- ✓ You just need to copy 4 objects from the sample programs provided to make "ImageCapture".
  - Let's modify Makefile
  - Combine CONNECT.CFG
  - Combine OBJECT.CFG
- ✓ Now 4 functions are combined!
- ✓ Summary: how to combine the modularized OPEN-R objects

# Combinação de Objectos PCI – Funções Desejadas

- ✓ After booting, AIBO stands up, and ceases any further movement.
- ✓ When its chin switch or back sensor is pushed, AIBO saves image data from its camera to the /OPEN-R/MW/DATA/P directory on the Memory Stick.

Data file names are

RGBH0000.BMP - RGBH9999.BMP	LAYER H the data is converted to BMP format
LAYH0000.RAW - LAYH9999.RAW	LAYER H raw data
LAYM0000.RAW - LAYM9999.RAW	LAYER M raw data
LAYL0000.RAW - LAYL9999.RAW	LAYER L raw data

The number included in the file name is increased consecutively.  
If a file already exists with that filename, the file isn't over-written.

- ✓ You can get image data via the wireless LAN; TinyFTPD is incorporated.
- ✓ If you want to shutdown AIBO, push the pause button.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 79

# Combinação de Objectos PCI – Estrutura Geral

**Move legs** → We are a group of operators who move AIBO. We are MoNet and friends of MoNet. We make AIBO stand after booting.

**ImageObserver** → I'm a famous camera-man ImageObserver. I will take nice pictures. "cheese"! Please push back sensor, and I will take a picture and save to the "Memory Stick".

**TinyFTPD** → I'm in charge of files. I will send files to the PC. I'm TinyFTPD. I will translate files in the "Memory Stick" to PC by ftp.

**Powermon** → I'm Powermon. I watch the power button. I will shutdown AIBO when the pause button is pressed.

**We are working independently**

6

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 80

# Combinação de Objectos PCI - Ficheiros Necessários

1. If you want to practice this tutorial, please rename original sample and make new folder.  

```
$ cd sample  
$ mv ImageCapture ImageCapture_org  
$ mkdir ImageCapture
```
2. Copy original ImageObserver to destination folder  

```
$ cp -r ImageObserver/ImageObserver ImageCapture/ImageObserver
```
3. Copy original MoNet/MoNetTest to destination folder  

```
$ cp -r MoNet/MoNetTest ImageCapture/MoNetTest
```
4. MS folder is based from MoNet/MS  

```
$ cp -r MoNet/MS ImageCapture/MS
```
5. Copy Passwd from TinyFTPD  

```
$ cp TinyFTPD/MS/OPEN-R/MW/CONF/PASSWD ImageCapture/MS/OPEN-R/MW/CONF/PASSWD
```
6. Copy Makefile from MoNet  

```
$ cp MoNet/Makefile ImageCapture/Makefile
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 81

# Combinação de Objectos PCI – Modificação da Makefile (MoNet)

```
COMPONENTS=ImageObserver MoNetTest ../MoNet/MoNet ../MoNet/MotionAgents ../MoNet/SoundAgent Y  
../PowerMonitor/PowerMonitor ../TinyFTPD/TinyFTPD
```

INSTALLDIR=\$(shell pwd)/MS  
TARGETS=all install clean

.PHONY: \$(TARGETS)

```
$(TARGETS):  
  for dir in $(COMPONENTS); do Y  
    (cd $$dir && $(MAKE) INSTALLDIR=$(INSTALLDIR) $*) Y  
  done
```

✓ And you need modify Makefile in MoNetTest (change the location of header file)

```
CXXFLAGS= Y  
-O2 Y  
-g Y  
-I. Y  
-I$(PREFIX)/OPEN_R/include/R4000 Y  
-I$(PREFIX)/OPEN_R/include Y  
-I../MoNet/common/include
```

You don't need modify the makefiles for the other objects!  
The installed location is specified by INSTALLDIR, so objects will be installed in the MS folder under MoNet.

Qualquer Makefile podia ser modificada mas esta exige apenas pequenas modificações

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 82

# Combinação de Objectos

## Construção do Ficheiro Connect.cfg

- ✓ Combine the CONNECT.CFG for MoNet, and the CONNECT.CFG of ImageObserver.
- ✓ TinyFTPD and PowerMonitor don't have CONNECT.CFG files, so you don't need to add them in the current CONNECT.CFG.

```
#
# MoNetTest <-> MoNet
#
MoNetTest.Command.MoNetCommand.S MoNet.ClientCommand.MoNetCommand.O
MoNet.ClientResult.MoNetResult.S MoNetTest.Result.MoNetResult.O
#
# MoNet <-> MotionAgents
#
MoNet.MotionAgentCommand.MoNetAgentCommand.S MotionAgents.Command.MoNetAgentCommand.O
MotionAgents.Result.MoNetAgentResult.S MoNet.AgentResult.MoNetAgentResult.O
#
# MoNet <-> SoundAgent
#
MoNet.SoundAgentCommand.MoNetAgentCommand.S SoundAgent.Command.MoNetAgentCommand.O
SoundAgent.Result.MoNetAgentResult.S MoNet.AgentResult.MoNetAgentResult.O
#
# MotionAgents --> OVirtualRobotComm
#
MotionAgents.Effector.OCommandVectorData.S OVirtualRobotComm.Effector.OCommandVectorData.O
#
# SoundAgent --> OVirtualRobotComm
#
SoundAgent.Speaker.OSoundVectorData.S OVirtualRobotAudioComm.Speaker.OSoundVectorData.O
#
# OVirtualRobotComm.FbkImageSensor.OFbkImageVectorData.S ImageObserver.Image.OFbkImageVectorData.O
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 83

# Combinação de Objectos

## Construção do Ficheiro Object.cfg

- ✓ Add new objects in your OBJECT.CFG (from the MoNet sample)

```
/MS/OPEN-R/MW/OBJS/POWERMON.BIN
/MS/OPEN-R/MW/OBJS/MTNAGTS.BIN
/MS/OPEN-R/MW/OBJS/SOUNDAGT.BIN
/MS/OPEN-R/MW/OBJS/MONET.BIN
/MS/OPEN-R/MW/OBJS/MONETEST.BIN
/MS/OPEN-R/MW/OBJS/IMAGEOBS.BIN
/MS/OPEN-R/MW/OBJS/TINYFTPD.BIN
```

} Objects of MoNet

- ✓ Don't forget, copy the PASSWD file from TinyFTPD /MS/OPEN-R/MW/CONF (in TinyFTPD) to /MS/OPEN-R/MW/CONF (in MoNet).

- ✓ You don't need to copy MONET.CFG and MONETCMD.CFG, as they are already in /MS/OPEN-R/MW/CONF of MoNet.

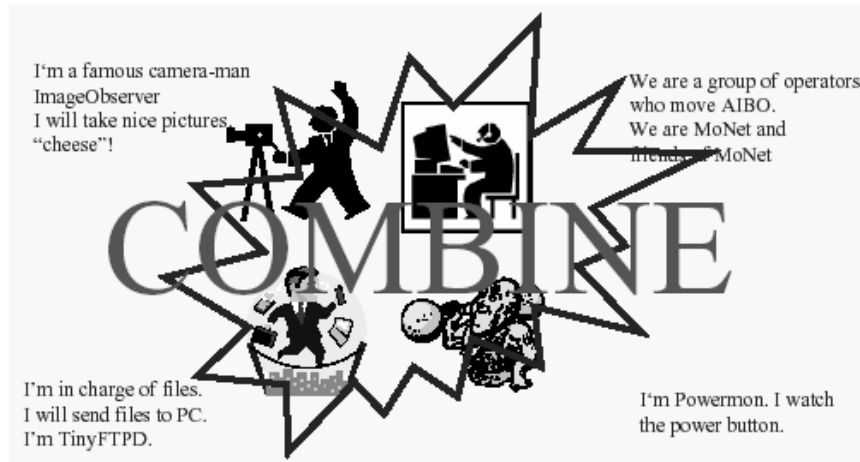
LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 84

# Combinação de Objectos

## PCI – 4 Funções Combinadas



LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 85

# Combinação de Objectos

## Sumário

- ✓ Gather each object in /OPEN-R/MW/OBJS
- ✓ Combine each CONNECT.CFG into one file (only add lines)
- ✓ Combine each OBJECT.CFG into one file (only add lines)

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 86

## Combinação de Objectos PCI – Modificações dos Objectos

- Modificar MoNet:
  - DoStart: Robô coloca-se em pé
  - Ready: Robô permanece em pé
- Modificar Image Observer:
  - Tirar fotografia quando o sensor do queixo é pressionado
  - Gravar todos os formatos de imagem e um BMP do formato – Layer H
  - Número dos ficheiros incrementado em sequência

LIACC-FEUP


Luís Paulo Reis / 2003

Slide Nº 87

## Combinação de Objectos PCI – Modificações dos Objectos

**Original**

MoNetTest object recognizes input from the console and then moves AIBO.



**After reconstruction**

After booting, AIBO moves its legs to the stand pose and keeps it in that pose.

```
MoNetTest.h
const MoNetCommandID SLEEP2SLEEP_NULL = 0; // see /OPEN-R/MW/CONF/MONHTCMD.CPG

MoNetTest.cc
OStatus
MoNetTest::DoStart(const OSystemEvent& event)
{
    if (subject [objCommand]->IsReady() == true) {
        Execute(SLEEP2SLEEP_NULL);
        moNetTestState = MNTS_WAITING_RESULT;
    } else {
        moNetTestState = MNTS_START;
    }
}

MoNetTest.h
const MoNetCommandID SLEEP2SLEEP_NULL = 0; // see /OPEN-R/MW/CONF/MONHTCMD.CPG
const MoNetCommandID STAND2STAND_NULL = 2; // see /OPEN-R/MW/CONF/MONHTCMD.CPG

MoNetTest.cc
OStatus
MoNetTest::DoStart(const OSystemEvent& event)
{
    if (subject [objCommand]->IsReady() == true) {
        Execute(STAND2STAND_NULL);
        moNetTestState = MNTS_IDLE;
    } else {
        moNetTestState = MNTS_START;
    }
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 88

# Combinação de Objectos

## PCI – Modificações dos Objectos

### Original

MoNetTest object recognizes input from the console and then moves AIBO.



### After reconstruction

After booting, AIBO moves its legs to the stand pose and keeps it in that pose.

```
MoNetTest.cc
void
MoNetTest::ReadyCommand(const OReadyEvent& event)
{
    if (moNetTestState == MNTS_START) {
        MoNetCommand cmd(SLEEP2SLEEP_NULL);
        subject[abjCommand] ->SetData(&cmd, sizeof(cmd));
        subject[abjCommand] ->NotifyObservers();
        moNetTestState = MNTS_WAITING_RESULT;
    }
}
```

```
MoNetTest.cc
void
MoNetTest::ReadyCommand(const OReadyEvent& event)
{
    if (moNetTestState == MNTS_START) {
        MoNetCommand cmd(STAND2STAND_NULL);
        subject[abjCommand] ->SetData(&cmd, sizeof(cmd));
        subject[abjCommand] ->NotifyObservers();
        moNetTestState = MNTS_IDLE;
    }
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 89

# Combinação de Objectos

## PCI – Modificações de Image Observer

### Original

1. Set color information to Cdt to recognize the color pink
2. Write data to "Memory Stick"; the order is LayerH, LayerM, LayerL, LayerC
3. After that, display the time data and color frequency information on the wireless console



### After our changes are applied

1. Take a picture and save it to "Memory Stick" when the chin sensor or back sensor is pressed
2. The saved images consist of raw data from LayerH, LayerM, and LayerL, as well as BMP format data of LayerH (which is converted from raw data).
3. The file name has a number and it increases sequentially; existing files are not over-written.

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 90

# Combinação de Objectos

## PCI – Dados dos Sensores do Queixo e Trás

- ✓ Invoke `OpenPrimitive()` in `DoInit()`
- ✓ In the `OpenPrimitive()` function, open each sensor and get its primitive ID
- ✓ The data can be retrieved in `Notify()`
- ✓ You can get the data by using `event.Data(0)`. If you want subsequent data, call `AssertReady()`

If you are unclear on these concepts, please review our previous tutorials.

**Let's learn how to get status of the chin sensor and back sensor !**

To get the status of a touch sensor, you can use a direct command to refer to them.

- ✓ To get status of a touch sensor use :  
`OPENR::GetSensorValue(PrimitiveID, OSensorValue* )`

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 91

# Combinação de Objectos

## PCI – Dados dos Sensores do Queixo e Trás

```
Initialize local value
ImageObserver::ImageObserver() : imageObserverState(IOS_IDLE),
                                fbkID(oprimitiveID_UNDEF), /* Image sensor */
                                tinswID(oprimitiveID_UNDEF), /* chin sensor */
                                backswID(oprimitiveID_UNDEF) /* back sensor */
{
}

OStatus
ImageObserver::DoInit(const OSystemEvent& event)
{
    NEW_ALL_SUBJECT_AND_OBSERVER;
    REGISTER_ALL_ENTRY;
    SET_ALL_READY_AND_NOTIFY_ENTRY;

    OpenPrimitive(); /* This is one pattern */

    return OSUCCESS;
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 92

# Combinação de Objectos

## PCI – Dados dos Sensores do Queixo e Trás

```
static const char* const FBK_LOCATOR = "PRM:/r1/c1/c2/c3/i1-FbkImageSensor:F1";
static const char* const TINSW_LOCATOR = "PRM:/r1/c1/c2/c3/c4/s5-Sensor:s5";
static const char* const BACKSW_LOCATOR = "PRM:/r6/s1-Sensor:s1";

void
ImageObserver::OpenPrimitive()
{
    OStatus result = OPENR::OpenPrimitive(FBK_LOCATOR, &fbkID);
    // The error process is omitted
    result = OPENR::OpenPrimitive(TINSW_LOCATOR, &tinswID);
    // The error process is omitted
    result = OPENR::OpenPrimitive(BACKSW_LOCATOR, &backswID);
    // The error process is omitted
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 93

# Combinação de Objectos

## Quando um dos Sensores é Pressionado Tirar uma Fotografia

```
void
ImageObserver::Notify(const CNotifyEvent& event)
{
    static int counter = 0;

    OFbkImageVectorData* fbkImageVectorData
        = (OFbkImageVectorData*)event.Data(0); // Here, get the picture

    if (imageObserverState == IOS_IDLE) {
        return; // do nothing
    }

    OSensorValue val1, val2;
    OPENR::GetSensorValue(tinswID, &val1);
    OPENR::GetSensorValue(backswID, &val2);
    if (val1.value == oswitchOFF & val2.value == oswitchOFF) {
        // If the back sensor is not pushed then skip and get next image
        goto ready;
    }

    while (ExistData(counter) == true) counter++; // If the file name exists, then skip the number
    SaveData(counter, fbkImageVectorData); // save the image to "Memory Stick"

ready:
    observer[event.ObsIndex()->AssertReady(event.SenderID());
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 94

# Combinação de Objectos

## Verificar se um Ficheiro Existe

```
bool
ImageObserver::ExistData(int serialNumber)
{
    char name[128];
    struct stat st;

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/RGBH%04d.BMP", serialNumber);
    if (stat(name, &st) == 0) return true;

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/LAYH%04d.RAW", serialNumber);
    if (stat(name, &st) == 0) return true;

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/LAYM%04d.RAW", serialNumber);
    if (stat(name, &st) == 0) return true;

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/LAYL%04d.RAW", serialNumber);
    if (stat(name, &st) == 0) return true;

    return false;
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 95

# Combinação de Objectos

## Gravar um Ficheiro

```
void
ImageObserver::SaveData(int serialNumber, OFbkImageVectorData* imageVec)
{
    char name[128];
    BMP bmp;

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/RGBH%04d.BMP", serialNumber);
    OSYSPRINT(("writing %s ..\n", name));
    bmp.SaveYCrCb2RGB(name, imageVec, ofbkimageLAYER_H);

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/LAYH%04d.RAW", serialNumber);
    OSYSPRINT(("writing %s ..\n", name));
    SaveRawData(name, imageVec, ofbkimageLAYER_H);

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/LAYM%04d.RAW", serialNumber);
    OSYSPRINT(("writing %s ..\n", name));
    SaveRawData(name, imageVec, ofbkimageLAYER_M);

    sprintf(name, "/MS/OPEN-R/MW/DATA/P/LAYL%04d.RAW", serialNumber);
    OSYSPRINT(("writing %s ..\n", name));
    SaveRawData(name, imageVec, ofbkimageLAYER_L);

    OSYSPRINT(("done.\n"));
}
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 96

# Combinação de Objectos

## Sumário

- ✓ The main feature of OPEN-R provides modularized software
- ✓ This is how to combine OPEN-R objects which are modularized:
  - Gather each object in /OPEN-R/MW/OBJS
  - Combine each CONNECT.CFG into one file (only add lines)
  - Combine each OBJECT.CFG into one file (only add lines)That's all!
- ✓ Invoke OpenPrimitive() in DoInit()
- ✓ In the OpenPrimitive() function, open each sensor and get its primitive ID
- ✓ The data can be retrieved in Notify()
- ✓ You can get data by using event.Data(0). If you want subsequent data, call AssertReady()
- ✓ To get status of touch sensors use :  
OPENR::GetSensorValue(PrimitiveID, OSensorValue\* )

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 97

## A Core Class

```
#include <OPENR/OObject.h>
#include <OPENR/OSubject.h>
#include <OPENR/OObserver.h>
#include "def.h"

class SampleClass : public OObject {
public:
    SampleClass();
    virtual ~SampleClass() {}

    OSubject* subject[numOfSubject];
    OObserver* observer[numOfObserver];

    virtual OStatus DoInit(const OSystemEvent& event);
    virtual OStatus DoStart(const OSystemEvent& event);
    virtual OStatus DoStop(const OSystemEvent& event);
    virtual OStatus DoDestroy(const OSystemEvent& event);

    //Describe the member functions corresponding to Notify,
    //Control, Ready, Connect method.
};
```

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 98

# Métodos Principais da Core Class

```
virtual OStatus DoInit(const OSystemEvent& event);
```

The DoInit() method is called at startup to initialize all the instances and variables.

```
virtual OStatus DoStart(const OSystemEvent& event);
```

The DoStart() Method executes right after DoInit() finishes execution.

```
virtual OStatus DoStop(const OSystemEvent& event);  
virtual OStatus DoDestroy(const OSystemEvent& event);
```

The DoStop() Method is called at shutdown. This is followed by DoDestroy() Method which destroys all the subject and observer instances, and calls the destructor function.

*LIACC-FEUP*

*Luís Paulo Reis / 2003*

*Slide Nº 99*

# Outros Métodos da Core Class

## Control Method

This method is used by the subject to receive the connection results with its observers.

## Connect Method

This method is used by the observer to receive the connection results.

## Notify Method

This method is used by the observer to receive a message from the subject.

## Ready Method

This method is used by the subject to receive the ASSERT-READY and the DEASSERT-READY notifications.

*LIACC-FEUP*

*Luís Paulo Reis / 2003*

*Slide Nº 100*

# Macros dos Métodos “Do”

## DoInit()

### NEW\_ALL\_SUBJECT\_AND\_OBSERVER

This registers the necessary number of subjects and observers.

### REGISTER\_ALL\_ENTRY

This registers the connection to services offered by other objects.

### SET\_ALL\_READY\_AND\_NOTIFY\_ENTRY

This registers all entry points.

## DoStart()

### ENABLE\_ALL\_SUBJECT

This enables all the subjects

### ASSERT\_READY\_TO\_ALL\_OBSERVER

This sends ASSERT\_READY to all subjects.

# Macros dos Métodos “Do”

## DoStop()

### DISABLE\_ALL\_SUBJECT

This disables all subjects of core class.

### DEASSERT\_READY\_TO\_ALL\_OBSERVER

This sends a DEASSERT\_READY to all connecting subjects.

## DoDestroy()

### DELETE\_ALL\_SUBJECT\_AND\_OBSERVER

This deletes all observer and subjects.

# Passagem de Mensagens

```
void SampleObserver::SendAssertReady()  
{  
    observer[obsFunc1]->AssertReady();  
}
```

```
void SampleSubject::Ready(const OReadyEvent& event)  
{  
    char str[32];  
    strcpy(str, "Some Text Message");  
    subject[sbjFunc2]->SetData(str, sizeof(str));  
    subject[sbjFunc2]->NotifyObservers();  
}
```

```
void SampleObserver::Notify(const ONotifyEvent& event)  
{  
    const char* text = (const char*)event.Data(0);  
    observer[sbjFunc1]->AssertReady();  
}
```

# Ficheiro Stub.cfg

```
ObjectName : SampleClass  
NumOfOSubject : 1  
NumOfOObserver : 2  
Service : "SampleClass.Func1.Data1.S", Control(), Ready()  
Service : "SampleClass.Func2.Data2.O", Connect(), Notify1()  
Service : "SampleClass.Func3.Data2.O", Control(), Notify2()
```

```
Extra : WakeUpOnLan()  
Extra : UpdateBatteryStatus()
```

## **ObjectName**

Name of the core class

## **NumOfOSubject**

Number of Subjects. Minimum of 1 subject is necessary. If there are no subjects needed you have to register a dummy subject.

## **NumOfOObserver**

Number of Observers. Minimum of 1 observer is necessary. If there are no observers needed you have to register a dummy observer.

**Service:**

This describes the various types of messages that is passed between the subject and its observers. The general format of a service is given here:

```
"(ObjectName).(Subname).(Data Name).(Service Type)",memb_func1(), memb_func2( )
```

**Subname**

This is a unique name for every service.

**Data Name**

The data-type used in the communication

**Service Type**

S (for subject) or O (for Observer)

**Memb\_func1()**

This function is called when a connection result is received. This function is implemented in the core class. If not needed, this entry can be "null".

**Memb\_func2()**

If the service is a service for observers, then this method is invoked whenever a message is received from the subject. If this is for the subject, this method is invoked each time the subject receives an ASSERT-READY or DEASSERT-READY from the observer. This function is implemented in the core class. If not needed, this entry can be "null".

**Extra**

This gives additional points of entry.

**LIACC-FEUP**

**Luís Paulo Reis / 2003**

**Slide Nº 105**

## Escrever o Ficheiro .ocf

This file has to be made prior to compiling and linking your files.

The file contains a single line of the format

```
object OBJ_NAME STACK_SIZE HEAP_SIZE SCHED_PRIORITY cache tlb user
```

OBJ\_NAME – This is the name of your object

STACK\_SIZE – This is the size of your stack in bytes. Be sure to allocate enough number of bytes

HEAP\_SIZE – This specifies the size of the heap in bytes.

SCHED\_PRIORITY – This denotes the scheduling priority. Set it to 128.

**LIACC-FEUP**

**Luís Paulo Reis / 2003**

**Slide Nº 106**

## Desenvolvimento de Programas

1. Projectar quais os objectos e suas funções
2. Decidir as estruturas de dados para a comunicação inter-objectos
3. Escrever o ficheiro stub.cfg
4. Escrever a classe “core” com as funções membro necessárias
5. Escrever o ficheiro connect.cfg e o ficheiro .ocf
6. Compilar e Linkar
7. Escrever os ficheiros object.cfg e designdb.cfg
8. Executar no AIBO
9. Teste e Debug

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 107

## Estrutura da Apresentação

- Introdução à Programação em OPEN-R SDK
- Comunicação entre Objectos
- Comunicação com os Objectos do Sistema
- Utilização da Câmara
- Combinação de Objectos
- Conclusões

LIACC-FEUP

Luís Paulo Reis / 2003

Slide Nº 108

# Conclusões

- “Samples” do OPEN-R SDK contêm:
  - Leitura de todos os sensores
  - Utilização de todos Actuadores
  - Comunicação Wireless
- Analisar, Compilar, Experimentar e Alterar as “Samples” fornecidas
- Combinar “Samples” para criar novos programas

Parte do Material Incluído nestes slides é copyright da Sony.  
Por favor não distribuir para fins comerciais estes slides.