GENOS – Protótipo de um montador de sistemas operacionais para sistemas embarcados

Filipe Renaldi

Antônio C. Tavares - Orientador

Roteiro

- Introdução
- Fundamentação Teórica
- Desenvolvimento do Trabalho
- Conclusão

Introdução
Motivação

- Interesse em sistemas operacionais e eletrônica;

- Sistemas embarcados em ascensão;

- Grupo de estudos em ARM na FURB.

Introdução
Objetivos do trabalho

Criar um ambiente de desenvolvimento de sistemas operacionais para sistemas embarcados utilizando componentes.

***Arquitetura ARM7.**

Fundamentação Teórica

Fundamentação Teórica

Sistemas Embarcados

Sistema Embarcado – Computador de propósito específico encapsulado no dispositivo ao qual ele controla.

Fundamentação Teórica
Sistemas Embarcados

- Sistema computacional com CPU, memória, periféricos;

- Projetado para uma função definida/ dedicada;

- Recursos limitados ao necessário.

Fundamentação Teórica
Sistemas Operacionais

Sistemas operacionais embarcados:

- Abstração do hardware;

- Separação do software de sistema e software aplicativo.

Fundamentação Teórica

Sistemas Operacionais

- Camada intermediária entre hardware e aplicativo;
- Consumir o mínimo de recurso possível;
- Estratégia de modularidade.

Sistema operacional de tempo real:

- *soft real- time:* restrição fraca de tempo;
- *hard real- time:* restrição forte de tempo.

Fundamentação Teórica
Componentes

Unidade de software que encapsula em si projeto e implementação oferecendo- o através de interfaces.

- **Interações somente através das interfaces;**
- **Interfaces oferecidas;**
- **Interfaces requeridas;**
- **Informações de propriedades não funcionais.**

Fundamentação Teórica
Componentes

- Redução de custo e tempo de desenvolvimento;
- Gerenciamento da complexidade;
- Desenvolvimento paralelo;
- Aumento da qualidade;
- Facilidade de manutenção.

Fundamentação Teórica
Arquitetura ARM

ARM7TDMI- S:

- **Vários fabricantes;**
- **Arquitetura RISC de 32 bits;**
- **2 conjuntos de instruções: 16 e 32 bits;**
- **Sem proteção de memória;**

Fundamentação Teórica
Arquitetura ARM

SoC LPC2106/ LPC2294:

- **Núcleo ARM7TDMI- S;**

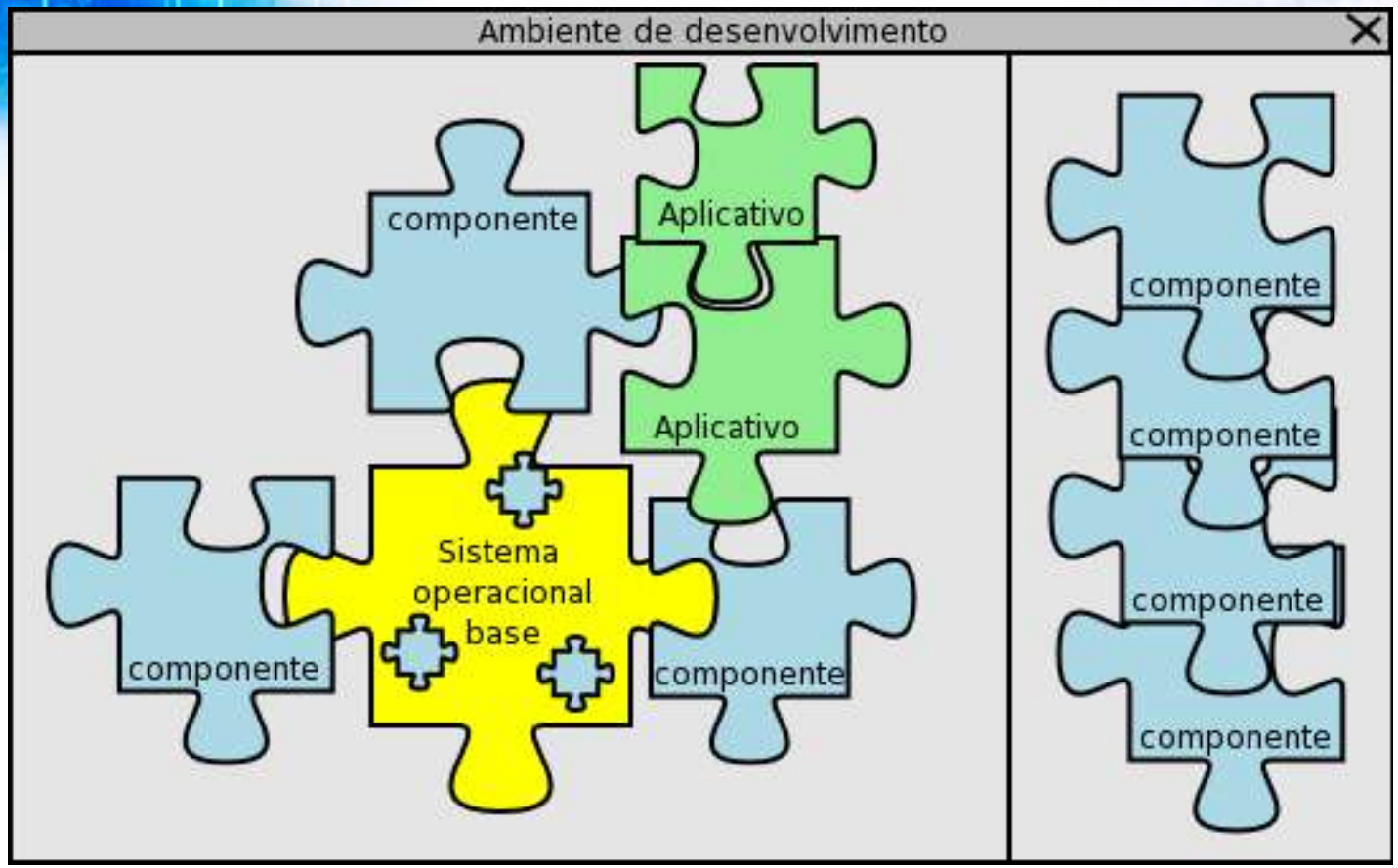
- **Vários periféricos: Timers, PWM, Com. Serial, RTC, GPIO, entre outros;**

- **Clock: 60Mhz;**

- **Memórias Flash e RAM.**

Desenvolvimento

Desenvolvimento
Visão da proposta



Desenvolvimento

GenosOS - Requisitos

- Núcleo multitarefa com suporte à prioridades;
- Controle de regiões críticas;
- Alocação de memória;
- Inicialização do sistema (SoC, Componentes, Aplicativo);
- Desenvolvido em linguagem C;
- Independente de SoC.

Desenvolvimento

GenosOS - Solução

Utilização do projeto FreeRTOS™

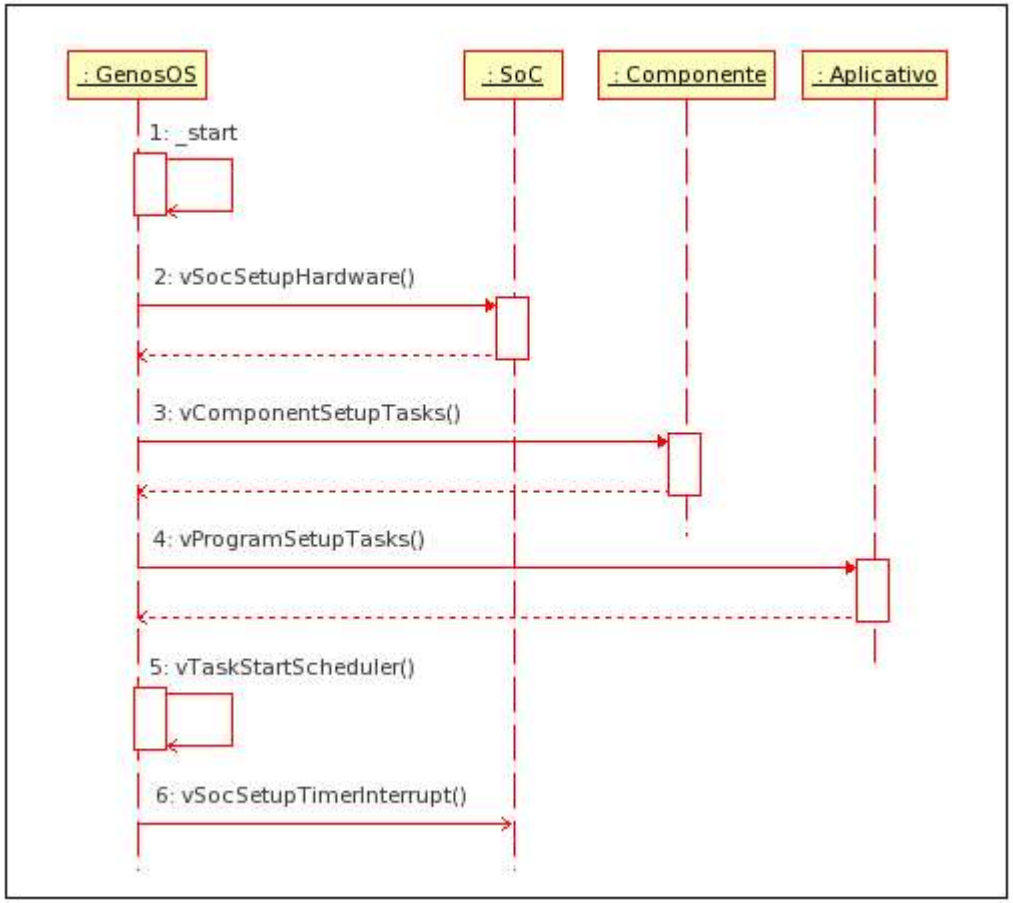
- Remoção do suporte a outros processadores;

- “Sabor” ARM7;

- Alteração na estrutura para adequar ao Genos.

Desenvolvimento

GenosOS - Seqüência



Desenvolvimento

GenosOS – Criação de tarefas

```
1   void minhaTarefa( void * parametros )
2   {
3       // Inicializações da minha tarefa.
4
5       for(;;)
6       {
7           // Faz alguma coisa
8       }
9   }
10
11
12  xTaskCreate(    minhaTarefa,            // Código da tarefa
13                  "nomeDaMinhaTarefa",    // Nome da tarefa
14                  0x400,                  // Tamanho da pilha da tarefa
15                  NULL,                   // Parâmetros
16                  3,                      // Prioridade
17                  handle );               // usado caso queira destruir a tarefa
```

Desenvolvimento
Componente

Componentes agregados ao GenosOS disponibilizam serviços às aplicações e outros componentes.

Objetivo:

- Modularizar e extender as funcionalidades do sistema;

- Reutilização de software: análise/ código/ testes.

Desenvolvimento
Componente - Requisitos

- Informações: Nome, versão, autor, SoC compatível e descrição;

- Interfaces oferecidas e interfaces requeridas;

- Documentação;

- Arquivos fontes em linguagem C.

Desenvolvimento

Componente - *component.conf*

```
1  [General]
2  Name=mensageiro
3  Description="Envia mensagens através de outro componente (serial, LCD, etc)."
4  ShortDescription="Componente que manda mensagens"
5  Arch=Arm7
6  Soc=*
7  Group=software
8  Author=Filipe
9  Version=1.0
10
11 [Api]
12 api0=enviaMsg(char *msg)
13
14 [Api-dependent]
15 api0=enviaString(char *str)
16
17 [Api-dependent-user]
18 api0=
19
20 [ArmFiles]
21 file0=mensageiro.c
22
23 [HeaderFiles]
24 file0=mensageiro.h
```

Desenvolvimento

Componente - Documentação

```
1  /**
2       \brief Envia um caracter pela porta serial
3       \param byte - Caracter a ser enviado
4       \return Sem retorno
5   */
6   void sendChar(char byte);
```

Funções

void sendChar (char *byte*)

Envia um caracter pela porta serial.

Parâmetros:

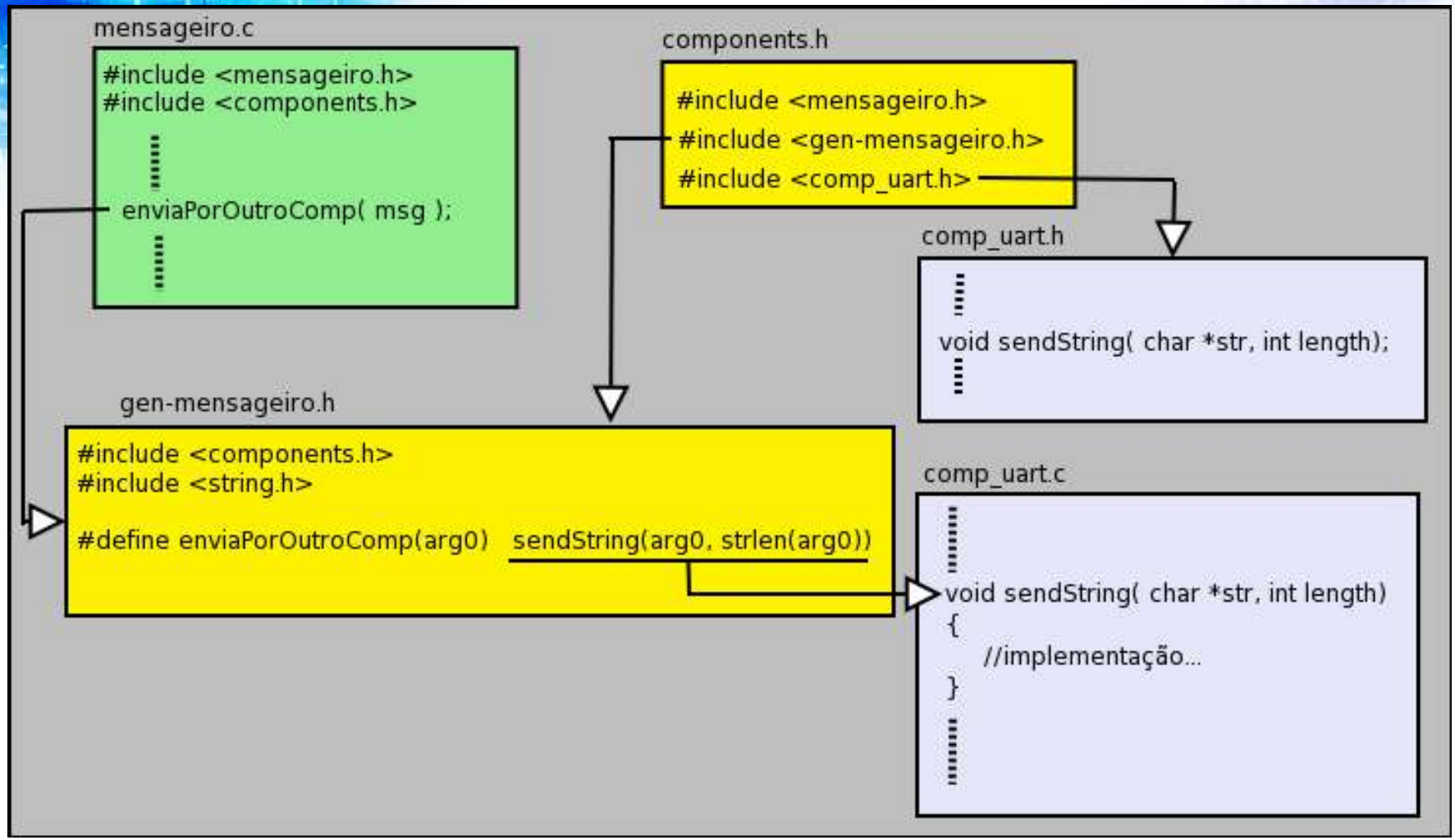
byte - Caracter a ser enviado

Retorna:

Sem retorno

Desenvolvimento

Componente - Normalização



Desenvolvimento
Componente - SoC

- Suporte ao modelo de SoC utilizado;

- Implementação das funções:

    - *void vSocSetupHardware(void);*

    - *void vSocSetupTimerInterrupt(void).*

- soc.h com definições de endereços dos registradores do SoC.

Desenvolvimento
Genos

Ambiente de desenvolvimento onde todas as partes são agragadas e oferecidas ao usuário.

“Área de trabalho do programador”

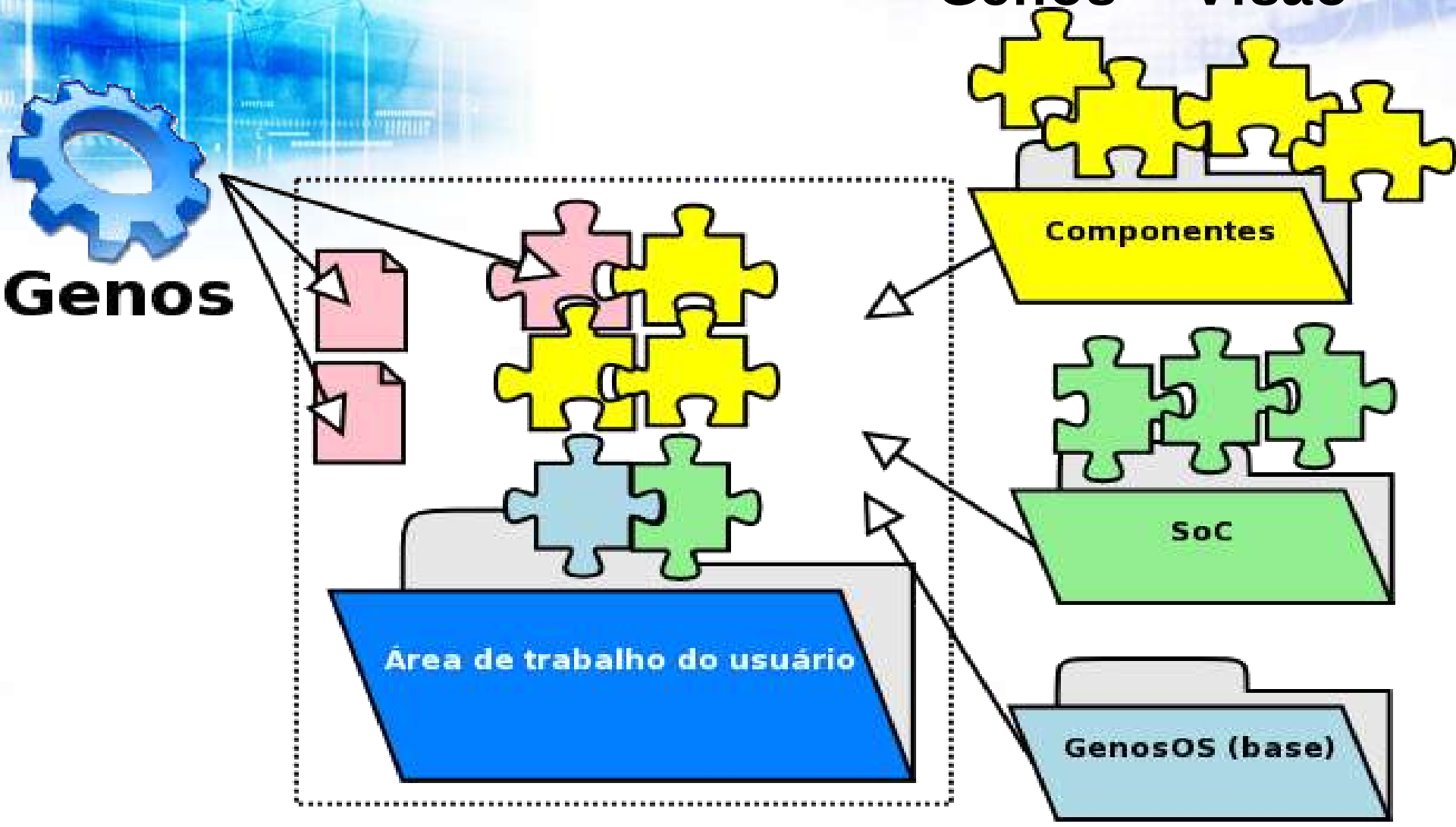
Desenvolvimento

Genos - Requisitos

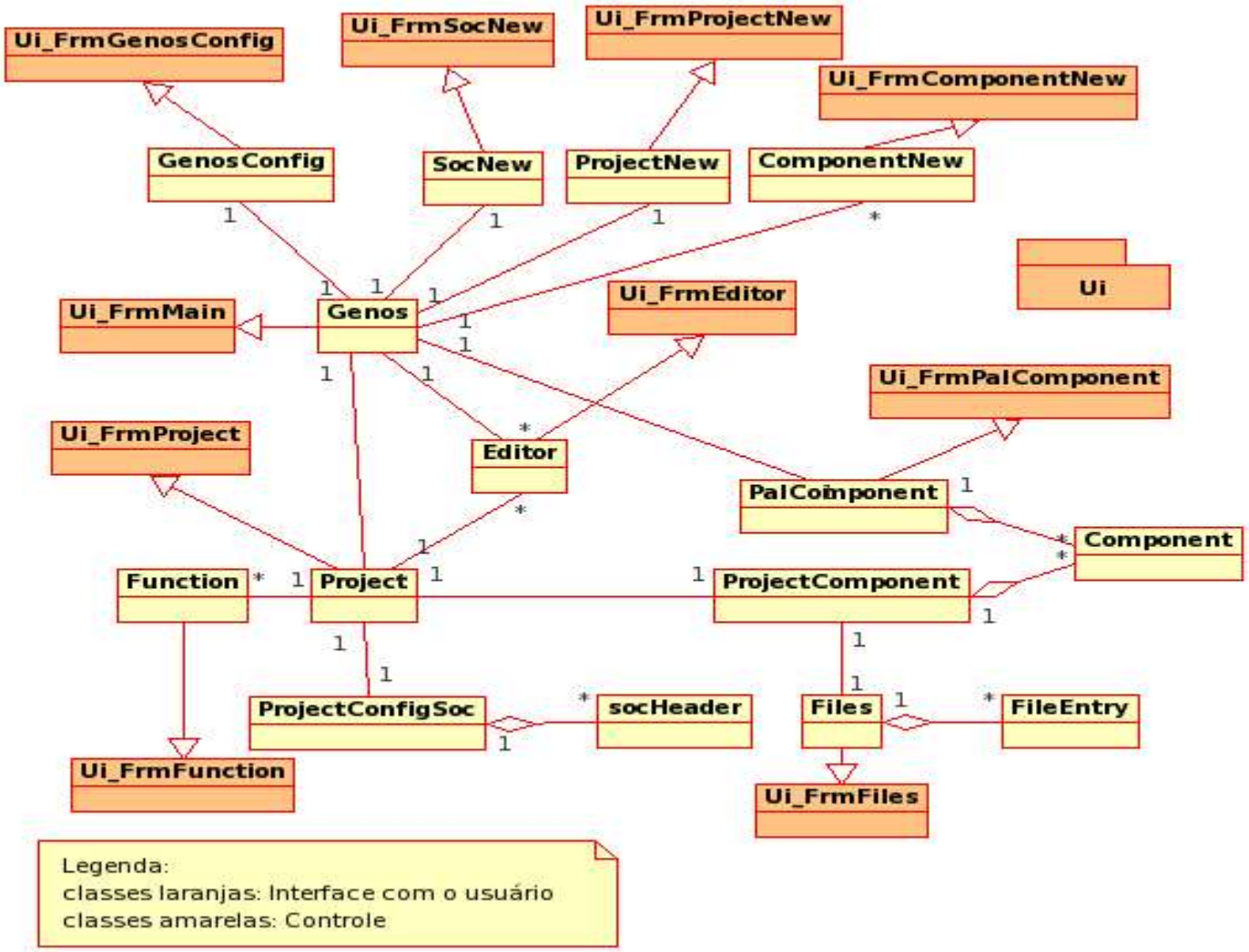
- Criação/ edição de SoC;
- Criação/ edição de Componentes;
- Criação/ edição de projetos:
    - Paleta de componentes;
    - Arquivos para a normalização;
    - Makefile e script.ld;
    - Geração da documentação;
    - Configuração dos componentes e parâmetros para o hardware;
    - Compilação do sistema;
- Editor de fontes integrado;

Desenvolvimento

Genos - Visão

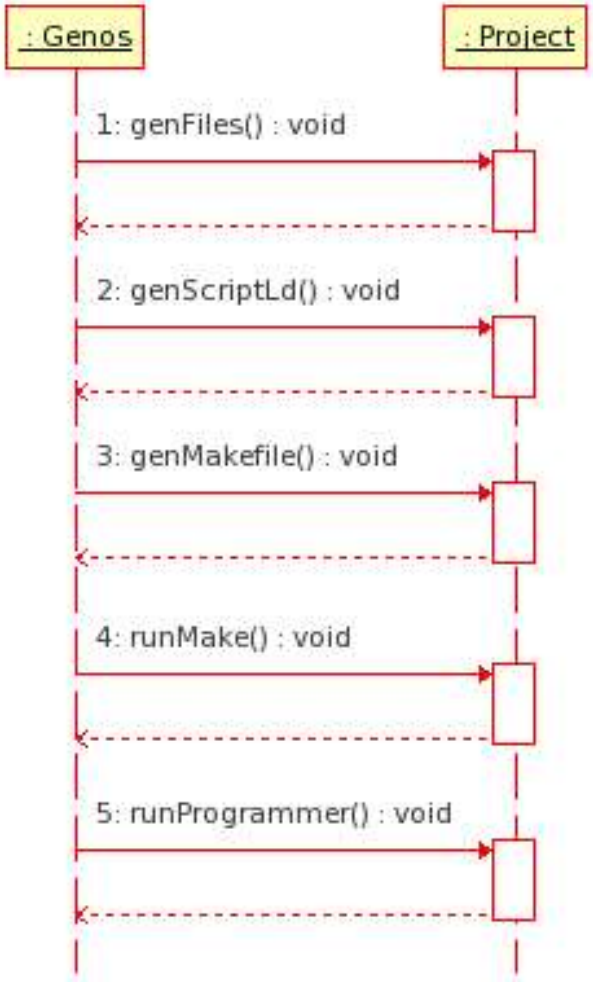


Desenvolvimento
Genos - Especificação



Desenvolvimento

Genos - Especificação



genFile() - criar os aquivos de normalização.

genScriptLd() - cria o script para a ferramenta de ligação a partir das informações de memória do projeto.

genMakefile() - criar o arquivo Makefile a partir de todos os arquivos do projeto, ferramentas do projeto e parâmetros do projeto.

runMake() - executa o comando make para compilar e ligar o código.

runProgrammer() - executa a ferramenta de programação configurada nas opções do projeto.

Desenvolvimento

Genos - Implementação

- Plataforma Linux (compatível Windows);
- Ambiente de desenvolvimento KDevelop;
- Linguagem C++ (compilador GCC);
- Ferramentas/ Classes de desenvolvimento QT;
- Ferramenta de especificação Umbrello;
- Controle de versões Subversion disponibilizado pelo Laboratório de Computação e Informática (LCI).

Resultados

- Testes efetuados com a placa Olimex- P2106 e Ulrich/ Miguel- LPC2294;

- Objetivos alcançados com o Genos gerando o binário do GenosOS;

- Modular e flexível;

- Ambiente de desenvolvimento suportando todos os estágios de desenvolvimento, inclusive o aplicativo.

Conclusão

Relevância em termos de conhecimento:

- Contribuição ao grupo de estudos de ARM da FURB;

- Perspectivas profissionais;

Trabalhos futuros

- **GenosOS multiplataforma;**

- **Componentes em formato binário;**

- **Suporte à métricas de tempo, para atender aos *hard real- time*;**

- **Criar uma modo de configuração da política de escalonamento.**

Demonstração...