

Echtzeitbetriebssysteme (am Beispiel QNX)

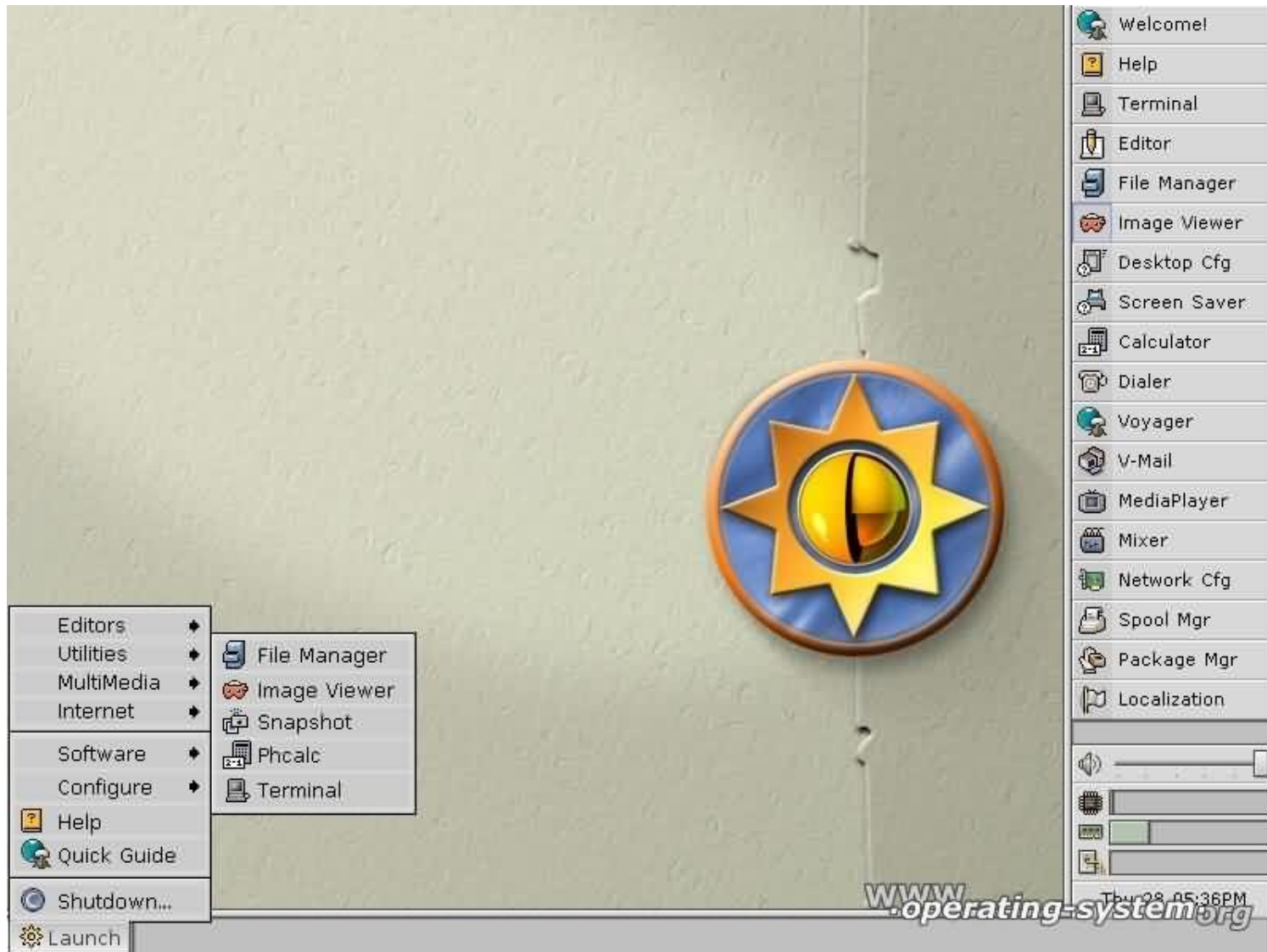
Dr. Stefan Enderle
HS Esslingen

2. QNX

2.0 Vorkenntnisse ?

- Unix / Linux
(Kommandos: ls, cd, rm, &, ...)
- C
- GNU GCC
(gcc, make)

2.1 Einführung



Hello World

- Terminal öffnen
- Editor öffnen: `ped hello.c`

```
#include <stdio.h>

int main ()
{
    printf("Hello World!\n");
}
```

- Kompilieren: `gcc hello.c`
- Ausführen: `./a.out`

Hello World

- Kompilieren mit anderem Namen:

```
gcc hello.c -o hello
```

- Ausführen: `./hello`

2.2 Threads

- Threads sind "leichtgewichtige" Prozesse, die vom Betriebssystem schnell zu starten und zu stoppen sind.
- Threads laufen parallel zum main-Prozess.
- Als Thread können Funktionen mit folgender Signatur gestartet werden:

```
void* <funktionsname> (void* <args>)
```

Wir verwenden meist:

```
void* <funktionsname> (void* not_used)
```

Threads erzeugen

```
#include <stdio.h>

void* thread1 (void* not_used)
{
    printf("thread1 running\n");
}

int main ()
{
    pthread_create (NULL, NULL, thread1, NULL);
    sleep(1);
}
```

Fragen

- Wer kommt wann dran ?
- Wenn der Thread1 eine Sekunde wartet, wer kommt dann dran?
- Und bei einem zweiten Thread?

Parallele Threads

```
#include <stdio.h>

void* thread1 (void* not_used)
{
    while(1) {
        printf("thread1 running\n");
        sleep(1);
    }
}

void* thread2 (void* not_used)
{
    while(1) {
        printf("thread2 running\n");
        sleep(3);
    }
}

int main ()
{
    pthread_create (NULL, NULL, thread1, NULL);
    pthread_create (NULL, NULL, thread2, NULL);
    sleep(15);
}
```