Tasking Framework in Nutshell: Motivation and Basic Features

Embedded Systems Week 2020
Olaf Maibaum
Small Satellite BIRD (Bispectral Infrared Detection)

- **BIRD**
  - Mass 94 kg
  - Size ~ 1x1x1 m
  - Launch 22.10.2001
  - Monitoring of high temperature events

- **Attitude Control System**
  - 3 axis stabilized
  - 9 sensors, 5 different types
  - 6 actuators, two different types
  - Distributed Kalman Filter
    - 14 Estimator Modules
    - 4 Predictor Modules

![Diagram of the BIRD Attitude Control System](chart.png)
Lessons learned BIRD Attitude Control System

- Controller use a predefined time schedule
- Hard real-time condition on reaction wheel commanding
- Complex data flow in Distributed Kalman Filter
- Slowest sensor determine start time of estimation, prediction and state controller
- Not enough computing time for:
  - Computation of high fidelity attitude state
  - Attitude state machine
  - Control torque computation
  - Reaction wheel command message preparation
- Deadline misses at times with high work load from other satellite bus applications
- Endless loops in thread bodies a nightmare for testing

→ Need for ASAP scheduling in Distributed Kalman Filter
→ No endless loops for computing tasks
ASAP Scheduling

Scheduling in BIRD:

Tasks with ASAP scheduling:

- Inspired by places (Channel) and transitions (Task) in Petri Nets
- Application SW overload Channel, Task, Group
- Class Channel control activations
- Global Scheduler
- Dynamic memory allocation

- First application in ATON (Autonomous Terrain-based Optical Navigation)
- Simulation experiment
- First call from S-function in MatLab/Simulink up to integrated in helicopter controller
- Application SW overload Channel, and Task
- Global Scheduler
- Dynamic memory allocation

- Drawbacks
  - Collision between timing MatLab/Simulink and pthread-Executors
    - Solution: Step processing
  - All channel activations becomes optional by multiple receiver tasks
  - Default behavior of Input and running condition in Channel::reset

- Application SW overload Channel, Event, Include, and Task
- Global scheduler with watchdog and reporting functionality based on Outpost
- Static memory allocation by memory provider class
- Specialized channels support PUS-Service 3 (Housekeeping)
- Specialized channels to support device specific functionalities
- Barriers introduced

- Drawbacks:
  - Global scheduler and unit tests by parallelity
    - Instrumentation of global scheduler for unit test support
    - Class UnitTestScheduler
      - to clean up global scheduler
      - Clock controlled by step processing
  - Memory provider class source of failures
  - Complexity to configure system by constructors

- Application on Distributed Systems
  - OBC-NG (On-Board Computer - Next Generation)
  - ScOSA (Scalable On-board Software Architecture)

- Application SW overload Channel, Include, Event, and Task
- Global scheduler on each computing node
- Static memory allocation by memory provider class

- Drawbacks:
  - Memory provider class source of failures
  - Complexity to configure system by constructors

• Requirements:
  • No global objects
  • Simplify set-up process
  • Low complexity of programming for platform specific schedulers
  • Overtake further lessons learned from applications

• By the requirements API needs an reorganization
  • Can not hold compatibility to old version
class API

Tasking::Task

Tasking::TaskImpl

Tasking::Input

Tasking::InputImpl

Tasking::InputArray

Tasking::InputArrayImpl

Tasking::Scheduler

Tasking::SchedulerImpl

Tasking::SchedulePolicy

Tasking::SchedulePolicyLifo

Tasking::Clock

Tasking::PeriodicSchedule

Tasking::PeriodicScheduleTrigger

Tasking::Channel

Tasking::Event

Tasking::EventImpl

Tasking::PeriodicScheduleImpl

Tasking::PeriodicScheduleTrigger
Software Architecture Changes

Tasking Framework in Nutshell

Olaf Maibaum  •  Embedded System Week 2020  •  20.09.2020

class API

Tasking::Task

Tasking::TaskImpl

Tasking::InputArray

Tasking::InputArrayImpl

Tasking::Input

Tasking::InputImpl

Tasking::Channel

Tasking::Event

Tasking::EventImpl

Tasking::Scheduler

Tasking::SchedulerImpl

Tasking::SchedulePolicy

Tasking::SchedulePolicyLifo

Tasking::Clock

PeriodicSchedule

PeriodicScheduleImpl

PeriodicScheduleTrigger

+ associatedScheduler

+ queueTail

+ scheduler

+ associatedTasks

+ clock

+ parent

+ policy

+ inputs

-m_channel

-m_task

+ activeTrigger

+ triggers

-impl

#head

+ parent

+ inputs

-impl

-m_inputs

Instantiation of scheduler in application software

class Scheduler

Tasking::Scheduler

Tasking::SchedulerUnitTest

Tasking::SchedulerExecutionModel

tp_numberOfExecutors:unsigned int

SchedulerPolicy:typename

Tasking::SchedulerProvider

Tasking::SchedulePolicy

-impl

+ parent

+ policy

Instantiation
Software Architecture Changes

Template classes to provide memory for management functionalities

- **Tasking::Task**
- **Tasking::TaskImpl**
- **Tasking::Input**
- **Tasking::InputImpl**
- **Tasking::InputArray**
- **Tasking::InputArrayImpl**
- **Tasking::Scheduler**
- **Tasking::SchedulerImpl**
- **Tasking::SchedulePolicy**
- **Tasking::SchedulePolicyLifo**
- **Tasking::Clock**
- **Tasking::PeriodicSchedule**
- **Tasking::PeriodicScheduleImpl**
- **Tasking::PeriodicScheduleTrigger**

Template classes:
- **Tasking::ConfigurationTemplates**
- **Tasking::InputArray**
- **Tasking::InputArrayImpl**
- **Tasking::InputArrayProvider**
- **Tasking::Group**
- **Tasking::GroupImpl**
- **Tasking::GroupProvider**
- **Tasking::SchedulerExecutionModel**
- **Tasking::SchedulerPolicy**
- **Tasking::SchedulerProvider**
- **Tasking::Task**
- **Tasking::TaskImpl**
- **Tasking::TaskProvider**
- **Tasking::TaskProviderImpl**
- **Tasking::TaskProviderImpl**

Structs:
- **Tasking::Input**
- **Tasking::InputImpl**
- **Tasking::InputArray**
- **Tasking::InputArrayImpl**
- **Tasking::InputArrayProvider**
- **Tasking::Group**
- **Tasking::GroupImpl**
- **Tasking::GroupProvider**
- **Tasking::Scheduler**
- **Tasking::SchedulerImpl**
- **Tasking::SchedulePolicy**
- **Tasking::SchedulePolicyLifo**
- **Tasking::Clock**
- **Tasking::PeriodicSchedule**
- **Tasking::PeriodicScheduleImpl**
- **Tasking::PeriodicScheduleTrigger**

Properties:
- **m_task**
- **m_inputs**
- **p_numberOfExecutors**
- **SchedulerPolicy**
- **numberOfInputs**
- **Policy**
- **clock**
- **parent**
- **queueHead**
- **queueTail**
- **activeTrigger**
- **triggers**
- **impl**
- **impl**
- **impl**
- **impl**
- **impl**

Methods:
- **+parent**
- **-impl**
- **-impl**
- **-impl**
- **+activeTrigger**
- **+triggers**
- **+head**
- **+parent**
- **+inputs**
- **-impl**
- **m_channel**
- **m_task**
Software Architecture Changes

- Inputs becoming part of class Task
- Inputs are synchronized by default
Software Architecture Changes

Configuration by methods

class API

Tasking::Task
-impl
+parent
+queueHead
-impl
-impl
-impl
-impl
+activeTrigger
+triggers
-impl
#head
+parent
+inputs
-impl
-m_inputs

Tasking::TaskImpl

Tasking::Input

Tasking::InputArray

Tasking::InputArrayImpl

Tasking::Scheduler

Tasking::SchedulePolicy

Tasking::SchedulePolicyLifo

Tasking::Event

Tasking::EventImpl

Tasking::Clock

Tasking::PeriodicSchedule

Tasking::PeriodicScheduleImpl

Tasking::PeriodicScheduleTrigger

Tasking::Scheduler

Tasking::SchedulerImpl

Configuration by methods

class ConfigurationMethods

Tasking::Input

+ associate(Channel&) : bool
+ configure(Channel&, unsigned int, bool) : void
+ configure(unsigned int, bool) : void
+ connectTask(TaskImpl&) : void
+ deassociate(void) : void
+ setName(char*) : void

Tasking::Task

+ configureInput(unsigned int, Channel&) : bool
+ setTaskId(TaskId) : void
+ setTaskName(char*) : void

Tasking::Group

+ join(Task&) : void

Tasking::TaskImpl

Tasking::EventImpl
Better separation between platform specific code and management methods
Software Architecture Changes

Better separation between API and internal interfaces
Software Architecture Changes

Better separation between platform specific code and management methods
Software Architecture Changes

Separate scheduling policies to make scheduler independent of it.
Software Architecture Changes

New class PeriodicSchedule to reduce clock queue length for strict periodic time lines

- Improve performance of clock queue for typical scenarios
- Bypass immediate events

New class PeriodicSchedule to reduce clock queue length for strict periodic time lines
Separate scheduling policies to make scheduler independent of it
Software Architecture Changes

```
class API
    Tasking::Task
        Tasking::TaskImpl
            Tasking::InputArray
                Tasking::Input
                    Tasking::InputArrayImpl
                        Tasking::InputImpl

    Tasking::Scheduler
        Tasking::SchedulerImpl
            Tasking::SchedulePolicy
                Tasking::SchedulePolicyLifo
                    Tasking::PeriodicSchedule
                        Tasking::PeriodicScheduleImpl
                            Tasking::PeriodicScheduleTrigger
                                +associatedScheduler
                                    +queueTail
                                        +scheduler
                                            +associatedTasks
                                                +clock
                                                    +parent
                                                        +periodicSchedule
                                                            +inputs
                                                                +m_channel
                                                                    +m_task
                                                                        +parent
                                                                            +inputs
                                                                                -impl

    Tasking::Channel
        Tasking::Event
            Tasking::EventImpl

    Barrier
        Tasking::Barrier
            Tasking::Channel
                Tasking::Barrier
                    Tasking::PeriodicScheduleTrigger
                        +activeTrigger
                            +triggers
                                -impl

    Group
        Tasking::Group
            Tasking::GroupImpl
                size:unsigned int
                    Tasking::GroupProvider
                        Tasking::TaskImpl
                            -impl
                                -taskList
                                    +group
                                        +taskList

```

The diagram shows the class relationships and their attributes, such as `+associatedScheduler`, `+queueTail`, `+scheduler`, `+inputs`, `+m_channel`, `+m_task`, `+parent`, and `+inputs`. The classes are interconnected to illustrate the software architecture changes.
Live Q&A session

Sunday 20.09.2020
at 11:00 – 12:00 am EDT
at 05:00-06:00 pm Berlin