UPVLC SSSA CTU CEA UC MNIS VT

OCERA

Information Society Technologies

OPEN COMPONENTS FOR REAL-TIME APPLICATIONS IST 2001 - 35102

http://www.ocera.org

Degraded Mode Management Framework

OCERA FT components are providing user support for the implementation of embedded real-time fault-tolerant applications. The Degraded Mode Management Framework has been designed to offer transparent management of dynamic reconfiguration of applications on detection of faulty situations. Continuity of service is maintained in case of partial failure through graceful degradation management.

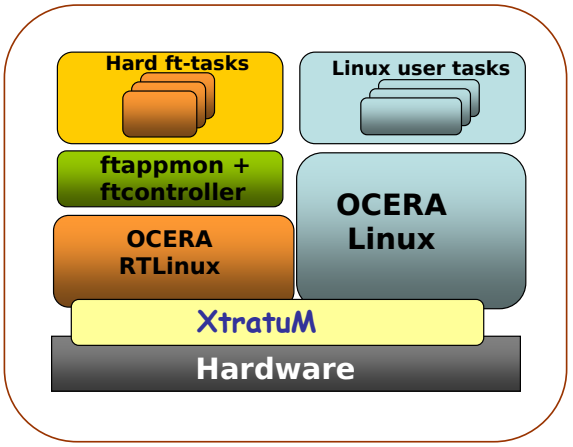
The Degraded Mode Management Framework offers an integrated set of tools and components.

❖ **Design/build tool**. The Ftbuilder permits the specification of application real-time constraints, different possible application modes along with related transition conditions.
From this specification, code generation is achieved in order to instantiate internal control data-bases of run-time components (ftappmon and ftcontroller) and to provide application model files.

❖ **API**. A specific but reduced API has been defined for manipulating so called ft-tasks. Three main functions: ft_task_init(), ft_task_create, ft_task_end(). These ft-tasks are actually encapsulation of periodic RTLinux tasks. Other functions are used to init internal data-bases. Besides these specific functions , application developers can use any RTLinux programming feature.

❖ **ftappmon** . The ftappmon component is devoted to global application handling. It is in charge of overall application setup and on reconfiguration decisions. It contains information on different possible application modes and on transition conditions. When an error is detected and notified by the ftcontroller, the ftmonitor analyzes the event and issues reconfiguration orders (stop, awake, switch ft-task behavior) towards the ftcontroller.

❖ **ftcontroller** . The ftcontroller is in charge of the direct control of application threads. It provides error detection (kill or timing error) and notification (towards ftappmon) and executes reconfiguration orders at task level.



Features

❖ RTLinux style programming (periodic threads)
❖ Error detection (kill, timing errors)
❖ Application modes handling
❖ Transparent dynamic reconfiguration on detection of abnormal situation.
❖ Ft-tasks have several behaviors (normal, degraded) predefined.

Model characteristics

❖ Periodic tasks
❖ Synchronized communication model
(one writer/several possible readers. A reader task reads values emitted during previous period of writer task).
❖Two possible behaviors (normal and degraded) for each task are defined by the developer. These behaviors are implemented in two alternative threads that are activated or suspended depending on application mode.

Performance

Hard real-time applications with 1ms period can be handled. For 20 ft-tasks with a total of cpu use of 70% application, reconfiguration can be achieved within the 1ms period (450 microseconds) in the worst case (all tasks change).

Universidad Politecnica de Valencia (Spain)
Czech Technical University (Cezch Republic)
UniControl A.S (Cezch Republic)
Visual Tools S.A. (Spain)
Scuola Superiore Santa Anna (Italy)
Commissariat a l'Energie Atomique (France)
MNIS (France)