International Journal of e-Collaboration
Volume 19 • Issue 1

SACRED:
Software Approach for Collaboration and Research Dissemination

Louella Colaco, BITS Pilani, Goa, India*
Arun S. Nair, BITS Pilani, Goa, India
Anurag Madnawat, BITS Pilani, Goa, India
Biju K. Raveendran, BITS Pilani, Goa, India
https://orcid.org/0000-0003-0749-0295

ABSTRACT

Collaborative research is an opportunity to bring creative minds together and blend multiple disciplines to churn out innovative solutions. In this era of massive social media and information overload, a streamlined process framework with best practices and procedures is a requirement for genuine scientific collaboration. The main aim of this work is to bring forth a software-centric framework for harmonizing research. SACRED is the outcome of experiences gained during the development of 'ARMS'-An Analysis Framework for Mixed Criticality Systems. ARMS is a collaborative platform to disseminate research and literature in real-time mixed criticality systems. ARMS is hosted on Amazon Amplify with the user interface implemented using the ReactJS framework. SACRED summarizes the software-centric process, practices and procedures followed, and renders it for similar collaborations in the future.

KEYWORDS

Collaborative Research Platform, Embedded Systems, Framework, Mixed Criticality Systems, Real-Time Systems, Software-Centric Process, Task Models, Task Simulators

INTRODUCTION

"None of us is as smart as all of us" - Kenneth Blanchard

The opportunity of connecting scientific and engineering minds in a collaborative fashion is a boon that yields stupendous results. In this era of information technology, collaborative research is feasible to a great extent as it brings together academicians and industrial engineers in smaller constellations. This provides multi-fold benefits to society and works as a stepping stone for budding researchers. The main challenges faced by a collaborative platform are: availability/approachability of domain

DOI: 10.4018/IJeC.315782 *Corresponding Author

expertise, proper division of labor among contributors, proper utilization of available skills, setting common goals among contributors, a common understanding among stakeholders, uniformity in work product, focused plan and regular updates to the community without losing zeal. In an effort to overcome these challenges, this paper presents a software approach that will harmonize research in a wider context. In other words, the work aims to bring the wealth of research contributions generated by diverse research groups under a common umbrella.

As part of this work, a group of mixed criticality systems (MCS) researchers launched a novel initiative platform - An Analysis Framework for MCS (ARMS), to build a large-scale research repository for enabling higher visibility of research results. ARMS aims at providing a framework to encourage collaboration between industry and academia thus enabling higher acceptance of research results by considering MCS as the test domain. It is a cloud-based platform designed for archiving, updating and reporting existing tasks models in MCS. The associated cloud-based database (DB) gets updated with on-going and new task models to keep up with recent and ever evolving research works. The ARMS platform compares and contrasts domain needs vis-a-vis with task models, attributes and presents a comprehensive landscape of the existing task model parameters in the mixed criticality domain. It also provides academicians and engineers the opportunity to choose task models appropriate to the specifics of the problem under study and provides a holistic view of chosen algorithms and models by analyzing, scheduling and generating statistics. The software-centric framework used in the development of ARMS is named - Software Approach for Collaboration and REsearch Dissemination (SACRED).

SACRED provides a detailed step-by-step software-centric approach that consists of three phases namely, setup, rollout and collaboration. The setup phase includes a systematic literature review, that results in identification, grouping and clustering of task parameters based on usage scenarios and functional attributes. It includes a multi-vocal literature review consisting of published works, grey literature and industrial contributions. The rollout phase launches the ARMS framework for stakeholder's use. It is a three-stage process, consisting of platform finalization, design & development and validation & deployment of the cloud-based framework. The collaboration phase provides facility for pursuits and distribution. It further helps in establishing partnerships that allow comparison of existing research results and facilitates in performing regular updates. The software-centric processes, practices and procedures followed in the development of ARMS is extensible to other domains/ disciplines wherever collaborative research yields benefits.

The paper is organized as follows: Section 2 presents related literature on collaborative platforms. Section 3 provides the software process followed by SACRED. Section 4 presents a case study of the SACRED process applied to the cloud-based aggregator platform ARMS. Section 5 lists the challenges faced/best practices followed in SACRED and a comparative analysis with other collaborative platforms and Section 6 concludes the work.

RELATED WORKS

Software development requires innovation, collective intelligence and collaboration of many minds. Collaborative platforms encourage association between industry and academia thus enabling dissemination of research results. A large number of collaborative platforms (Beck et al., 2022; Borgho & Teege, 1993; Brownson et al., 2021; Chorfi et al., 2022; Gesing et al., 2019; Khan et al., 2021; Lautamäki et al., 2012; McLennan & Kennell, 2010; Monnard et al., 2021; Stodden et al., 2012) have been presented in literature over the years. HUBzero (Gesing et al., 2019; McLennan & Kennell, 2010) is one such platform that allows researchers to collaborate and network to develop simulation/modeling tools. These tools are made accessible to the community through the web browser and launched in a grid infrastructure. Borgho and Teege (1993) presented a collaborative editor for managing software engineering projects. The editor provides facility for distributed editing, notifications and imparts consistency through dynamic voting. A web based java editor was proposed by Lautamäki et al. (2012) to

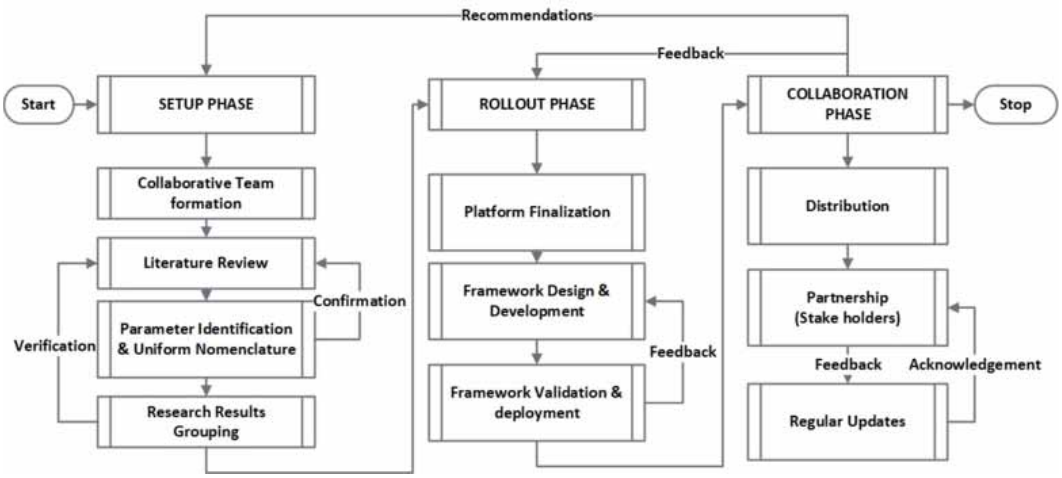
support collaboration for development of java applications. The online editor provided support for error detection, automatic code generation and social media features. Another collaborative platform to enhance programming skills was presented by Chorfi et al. (2022). Geographically distributed learners were able to interact with one another and with mentors to solve problems and develop shared programs. Beck et al. (2022) proposed an interdisciplinary collaborative research framework to disseminate innovations in the field of science. An approach to disseminate research on public health among the general and research community was proposed by Monnard et al. (2021). Mentored training for distributing cancer research is discussed in Brownson et al. (2021). Khan et al. (2021) studied the adoption of collaborative learning through social media during the Covid-19 pandemic. The work aimed at understanding the correlation between student performance and social media use, to comprehend the impact of social media on students during the pandemic. Stodden et al. (2012) pointed out that the research results presented in various works are unavailable for use by the research community. Their work presented a cloud based infrastructure that provided support and access to both the code and results from various published articles. Users were also able to experiment using their own data. The work by Peng et al. (2014) summarizes and compares software collaborative platforms in terms collaboration, co-ordination, awareness, communication and value transfer.

Based on the literature survey, it is observed that there are limited collaborative research dissemination platforms for the real-time systems community. The scope of ARMS and the corresponding software process - SACRED is to build a software-centric design level task modelling and scheduling framework of real-time MCS. It also assists the research community to understand the state-of-the-art research in the domain and evaluates by comparing with existing solutions thus allowing to propose and publish new ideas.

SACRED OVERVIEW

Collaborative research requires a framework as well as a process that describes how things are executed with a focused approach towards improvement. It is also important that this process provides a starting platform for budding researchers. Situations like the pandemic or other calamities hamper personal interactions and collaborations. But technology helps to overcome such situations by bringing about a holistic platform where researchers with a common mindset can collaborate and excel together. SACRED provides a detailed step-by-step approach to make a simple collaborative research framework

Figure 1. SACRED- process steps



and provides a cloud-based platform for researchers' use. This platform development requires a distinct mechanism rather than following well-known methodologies like verification and validation cycle, waterfall or agile model as it focuses on software research, collaboration, coordination, awareness, communication and value transfer.

SACRED consists of a three-phase software centric approach that includes setup, rollout and collaboration phase as depicted in Figure 1 . As part of the setup phase, the researcher starts with a systematic literature review, that extends to aid in identification of parameters and subsequently helps to list out a uniform nomenclature by assimilating the commonalities among these parameters. Further, these parameters are grouped based on domain specific criteria. The rollout phase includes platform finalization and the design and development of the cloud-based framework for easier access to the research data. The next phase - collaboration - provides facility to compare existing research results and makes it feasible to compare one's own results with the results available in the research fraternity. It also allows industrial and academic contributions by researchers. This platform serves as a stepping stone as well as a decision support system for existing researchers, designers and budding engineers. The following subsections describe the phases with emphasis on the expected outcomes of each phase and the best practices used.

Setup Phase

The purpose of this initial phase is to form a collaborative working group consisting of researchers, developers and collaborators and to prepare guidelines for regular communication and day-to-day activities. This step also elaborates on activities like literature review, preparation of common guidelines, work instructions, checklist and templates. Based on the domain of study, it is feasible to come up with a common framework or grouping of state-of-the-art research results. Researchers and developers have to also factor in the available time constraints and decide on the resource requirements.

The expected outcomes after the successful completion of this phase include:

- Formation of an active core team - Prior to starting the literature review, a team comprising of researchers, developers and collaborators who will aid in the review, development and updating needs to be put in place. All this is achieved through brain storming and targeted discussions with like-minded people which will provide the needed momentum for subsequent stages of research.
- Guidelines, checklists, templates, communication framework and systematic literature review - This step aids in preparing a common process and communication framework for working together. It includes guidelines for information exchange, regular connects and meeting agenda. It also includes preparation of checklist and templates to gather information and review of contents. An important engineering step that forms the basis of any research is a systematic literature survey. This includes collecting existing literature in a specific area and analyzing it to identify gaps and feasible improvements required in the domain of study.
- Parameter Identification with Uniform Nomenclature - Based on the literature review, high level parameters are extracted and classified. The parameter extraction process involves the identification and extraction of relevant parameters from the selected papers. These parameters may be represented using different notations by various researchers. There is a need to consolidate the parameters and provide a uniform nomenclature (including the notations) to make it usable by researchers in the relevant domain.
- Grouping/clustering state-of-the-art research results- Classification of research results into broad groups based on clusters in the landscape or based on industrial needs is an essential step. The parameters identified during the identification phase are grouped based on different criteria.

The best practices established during this phase include preparation of work instructions by each and every team member, establishment of the frequency and mode of communication, weekly report submission on progress and risks, template preparation and guidelines for capturing information, regular presentations and demos among team members.

Rollout Phase

The steps in the rollout phase aim to provide a dissemination framework based on the respective domain. It includes selection of the rollout platform, design, development and validation of tools. In this framework, codes of algorithms and models of the existing works are tested. The authors of the publication are contacted for the code base. If it is provided by them, the development team incorporates the same in the framework. If the authors provide only the execution permission to access their servers with the help of an interface, then the framework incorporates the work with the help of foreign servers. In the worst-case, the developer team codes the algorithms and sends them to the authors for correctness test before incorporating the same in the framework.

The expected outcomes after the successful completion of this phase are:

- Platform finalization - Platform finalization is the key to providing a user friendly and multi-platform experience. For the MCS field of research, this work finalized a web/mobile based application where the MCS research community can interact with the system through an app or web browser. The platform is deployed on a cloud server (Amazon Web Services (AWS)) for seamless accessibility and security.
- Framework design and development - Different methods can be used for aggregating and summarizing findings of various works. For the MCS task model work, a cloud-based aggregator is designed and developed. This aggregator is a client-server platform which will help researchers in identifying the most appropriate task models for their work. This framework archives the state-of-the-art task models and task parameters in MCS and provides various options for the researchers to choose task parameters in terms of various usage scenarios and functional attributes.
- Framework validation and deployment – A deployed framework needs to be tested with published data and validated with the authors wherever possible before rolling out to collaborators. In this step, help can be taken from volunteers in the research community to support the process.

The best practices implemented at this stage include object oriented modularized design with usage of formal modelling languages like Unified Modeling Language (UML). Collaborative testing by peers, like- minded researchers and authors improves the quality and user friendliness of the framework.

Collaboration Phase

The purpose of this step is to collaborate with an extended groups of researchers and yield results as an authentic freeware platform. The expected outcomes after the successful completion of this phase are:

- Distribution - Distribution, deployment and maintenance is carried out in this phase. The freeware version of the framework should be made available to users to decide and finalize on appropriate choices for their work. For example, ARMS framework will be administrated and maintained by a group formed by MCS researchers. Controlled access will be provided to users upon on-demand request.
- Research assimilation, decision support system & partnership - Decision support systems aid in qualitative analysis. In this work, to facilitate decision making, the task model framework is extended with scheduling algorithms to perform schedulability analysis. This will aid designers in performing comparative analysis and identifying apt task models for their study. This step also allows integration of author's code base / execution on their server / review of newly developed code base by the authors.
- Updates - Regular and streamlined updates are vital in order to add new features or improve existing ones. Regular updates with industrial and academic contributions on task models and schedulers makes the tool a robust guide to researchers and engineers thereby facilitating mutual benefits.

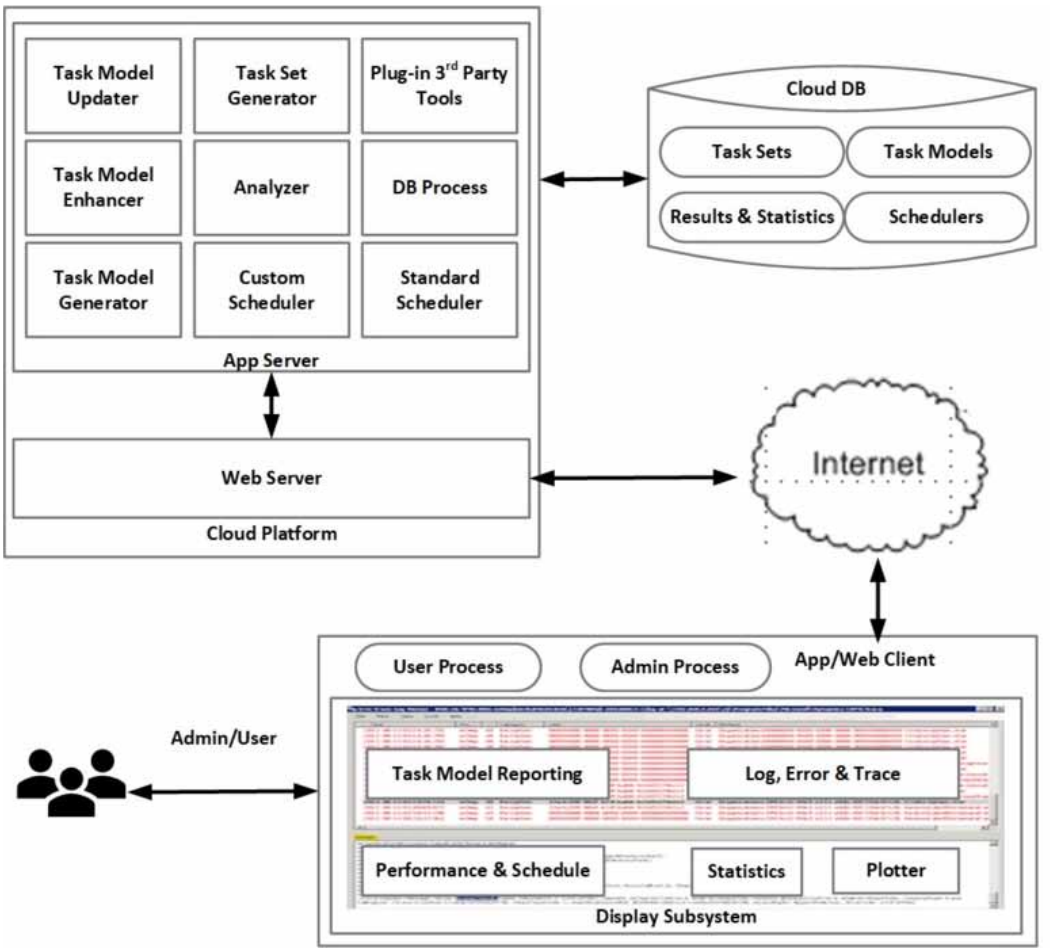
Swift and speedy responses among collaborators and their active participation to contribute latest research and artifacts make the framework up-to-date, relevant and purposeful.

AN ANALYSIS FRAMEWORK FOR MCS (ARMS) IMPLEMENTATION - A CASE STUDY

This section describes a factual experience that triggered the SACRED process methodology. It is the development of a platform for mixed criticality researchers. This platform follows a cloud-based client-server architecture and helps researchers in identifying the most appropriate task models for their applications along with detailed analysis. In this platform, the state-of-the-art task models and task parameters in MCS are archived. The aggregator tool platform generates task sets based on the chosen task model parameters and performs schedulability analysis. It facilitates researchers and engineers to add or enhance task models and scheduling algorithms. Regular bi-annual updates with industrial and academic contributions make this platform a robust guide to researchers and engineers thereby facilitating mutual benefits.

The typical architecture of the task model aggregator platform is depicted in Figure 2 . The app server consists of seven software modules - Task Model Updater, Task Model Enhancer, Task Model

Figure 2. Block schematic of the aggregator platform



Generator, Task Set Generator, Analyzer, DB Process and Scheduler. The web client/app at the user end consists of a User Process, Admin Process and a Display Subsystem. Major functionalities of this aggregator platform include task model generation, task set generation with selected task parameters, scheduling of task sets with mixed criticality schedulers and detailed analysis of results. The list of features supported are:

- Administered addition of task parameters and task models based on recent publications of repute.
- Collaborative maintenance framework for addition/deletion/enhancement of task models, parameters and usage scenarios.
- Selection of task models and task parameters based on nature of study and usage scenarios.
- Selection of scheduler from the standard set and feasible enhancement with customized or third-party plug-in schedulers.

Setup Phase

As the first step, a team was setup for literature review and development. The team comprised of a team lead, researchers and developers. The team was augmented with a sustenance consortium to nurture and maintain the aggregator platform. The group of researchers conducted a systematic literature review on task models in the MCS domain. The review covered 15 years of MCS research which started with the pioneering work by Vestal (2007) in 2007. After elaborate discussions, brainstorming sessions and several iterations more than seventy task model parameters were identified. Few of these parameters are listed with uniform nomenclature in Table 1. Further, these parameters were analyzed and grouped based on hardware configurations, usage scenarios and functional attributes. The hardware configurations include uni-core and multi-core processor types. The usage scenarios are resources, quality of service (QoS), context switching, energy efficiency, fault tolerance and parallel processing. The functional attributes are defined with respect to each usage scenario. As an example, task parameters required for modelling MCS with the resource usage scenario for both uni-core and multi-core systems can be categorized into three main functional attributes namely, shared resource usage, resource synchronization and communication (message passing). When considering shared resources such as memory, the basic task model (Vestal, 2007) is extended with parameters like minimum/maximum number of memory accesses (Pellizzoni et al., 2010), worst case number of cache misses (Yun et al., 2012), worst case memory access time (Li & Wang, 2016), worst case number of L1/LL cache misses (Nair et al., 2019), number of memory accesses (Awan et al., 2018) and intra/inter-core blocking times (Burns, 2013; Nair et al., 2019). Resource synchronization includes parameters like blocking times (Burns, 2013), priority (Zhao et al., 2014), active criticality (Zhao et al., 2014) and preemption level (Zhao et al., 2015) to deal with issues of priority and criticality inversion. The parameters considered in communication are message size (Tamas–Selicean & Pop, 2011), release time jitter (Burns & Davis, 2013) and worst case communication time (WCCT) (Dridi et al., 2019).

Operating system (OS) overheads due to context switching has substantial impact on the schedulability of the system. The task parameters that contribute to context switching include priority, address space and context switching times before and after execution of the task in each mode of operation (Davis et al., 2018; Evripidou, 2016). In MCS, increasing energy requirements demand efficient energy optimization techniques. This necessitates amendments in task modelling related to energy parameters like energy/power consumption vector (Awan et al., 2016), processor frequency levels (Huang et al., 2014), voltage and frequency scaling factors (Taherin et al., 2018). In order to achieve fault tolerance, parameters like failure probability (Guo et al., 2015), worst case execution time (WCET) of backups (Pathan, 2014), replication/distribution (Thekkilakattil et al., 2014), reliability constraints, dropping factor and various overheads (Choi et al., 2018) are considered. QoS is a mechanism to improve the schedulability of low criticality tasks. It is a feature that needs

to be incorporated while considering all other aspects such as energy, resources, fault tolerance, OS overheads etc. The parameters that contribute to this feature include importance (Fleming & Burns, 2014), tolerance limit (Gu et al., 2015), WCET in degraded mode (Giannopoulou et al., 2013), QoS values (Pathan, 2017), etc.

The introduction of multi-core architectures brings ample opportunities to implement parallel processing. Parallel processing systems consider parameters like number of threads (Liu et al., 2014), number of cores (Gill et al., 2018) and work and span for parallel tasks (Gill et al., 2018). Graph based task models are well suited to illustrate inter/intra task dependencies. Mixed criticality applications represented as graphs have vertices and edges. Edges can be classified as control flow edges and mode switching edges (Ekberg et al., 2013). Some of the parameters associated with graphs include mode of the job (Ekberg et al., 2013) and a function that defines the interference among tasks (Huang et al., 2013). Probabilistic task models consider parameters of execution time probability mass function (B. Alahmad et al., 2011), execution demand random variables and measure of probability (B. N. Alahmad & Gopalakrishnan, 2018).

The research team leader reviewed and accepted the task parameter grouping and confirmed the authenticity of the task models. The swimlane for this process is shown in Figure 3.

Rollout Phase

ARMS is a client-server web/mobile app that aims to provide an easy-to-use decision support system based on contemporary research results in MCS. The ARMS aggregator tool was hosted on Amazon Amplify (Version 4.43.0). The user interface was implemented using ReactJS 17.0.1. The DB is managed using Amazon DynamoDB and communication with the DB is handled using GraphQL application programming interfaces (APIs). Amazon S3 storage bucket and Amazon Cognito are used for storage and controlled access respectively.

Based on user preferences all task models that match the user selection criteria are displayed. The user selects one of the displayed task models for further analysis. The swimlane for task model generation, scheduling and analysis is shown in Figure 4. Upon selection of a model, the user can choose task parameters and retrieve task sets. The user is provided with facility to customize task parameter values before invoking the scheduler. The scheduler service, standard or custom, is used for scheduling and analyzing task parameters. The user is allowed to choose from the standard list of existing schedulers for analysis. They also have options of using customized schedulers or third-party solutions for scheduling task sets. The results of the scheduler are used for detailed analysis. The sequence diagrams for these processes are devised. Figure 5 shows the task model and task set generator sequence diagram. Here, the user selects attributes and usage scenarios. It views all the appropriate task models based on its selection. The user then selects a task model and chooses parameters. These values are used for the generation of sample task sets. The user is allowed to customize these task sets on a limited scale. These task sets are sent to the scheduler that generates schedules and statistics. Figure 6 shows the sequence diagram for scheduling and statistics generation. The task set customized by the user is used for scheduling. The user selects the appropriate scheduling mechanism. If the user

Figure 3. MCS aggregator - setup phase

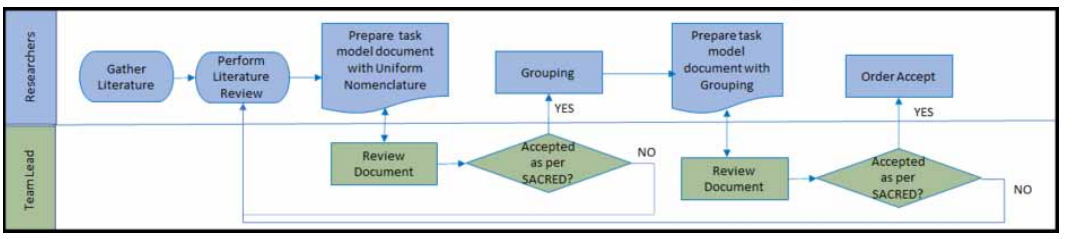


Table 1. Task parameters for MCS

Symbols	Interpretation	References
Resource Parameters		
$\overline{b}$ /B	Intra/Inter-core blocking times	Nair et al., 2019
λ	Preemption level	Zhao et al., 2015
$\overline{\mathrm{J}}$	Release time jitter	Burns and Davis, 2013
$\wedge min$, $\wedge max$	Minimum / Maximum number of memory accesses	Pellizzoni et al., 2010
Q/$\overline{Q}$	Number of memory accesses	Awan et al., 2018
$\overline{C}$	Worst case communication time	Dridi et al., 2019
Context Switching Parameters		
$_{c}cs$	Context switching time	Davis et al., 2018
A	Address Space	Davis et al., 2018
Energy Parameters		
E/$\vec{E}$	Energy/Power Consumption	Awan et al., 2016
ρ	Voltage and frequency scaling factor	Taherin et al., 2018
Fault Tolerance Parameters		
f	Failure probability,	Guo et al., 2015
B	WCET of Backups	Pathan, 2014
m	Replication requirement	Thekkilakattil et al., 2014
QoS Parameters		
I	Importance	Fleming and Burns, 2014
TL	Tolerance Limit	Gu et al., 2015
V / $\overline{V}$	QoS Values	Pathan, 2017
Parallel Task Parameters		
M	Number of cores	Gill et al., 2018
$\overrightarrow{work}$	Work for parallel tasks	Gill et al., 2018
$\overrightarrow{span}$	Span for parallel tasks	Gill et al., 2018
Probabilistic Task Parameters		
$\overrightarrow{fe}$	Execution time probability mass function	B. Alahmad et al., 2011
Ψ	Execution demand random variable	B. N. Alahmad and Gopalakrishnan, 2018
P	Probability measure	B. N. Alahmad and Gopalakrishnan, 2018
Graph Based Parameters		
^{e}cf	Directed edge that defines task control flow	Ekberg et al., 2013
^{e}ms	Directed edge that defines mode switch	Ekberg et al., 2013
ς	Function of allowed interference between tasks	Huang et al., 2013

prefers standard scheduler, the schedules and statistics generated by this scheduler are displayed to the user. If the choice is a custom scheduler, the user creates and uploads the same. The schedules

Figure 4. Task model generator and analyzer swimlane

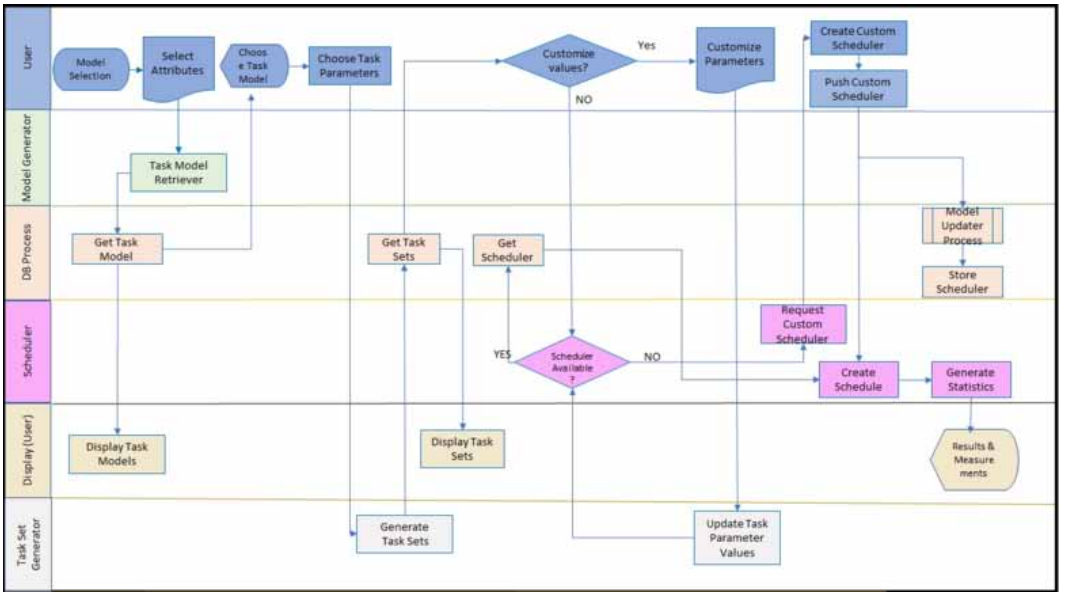
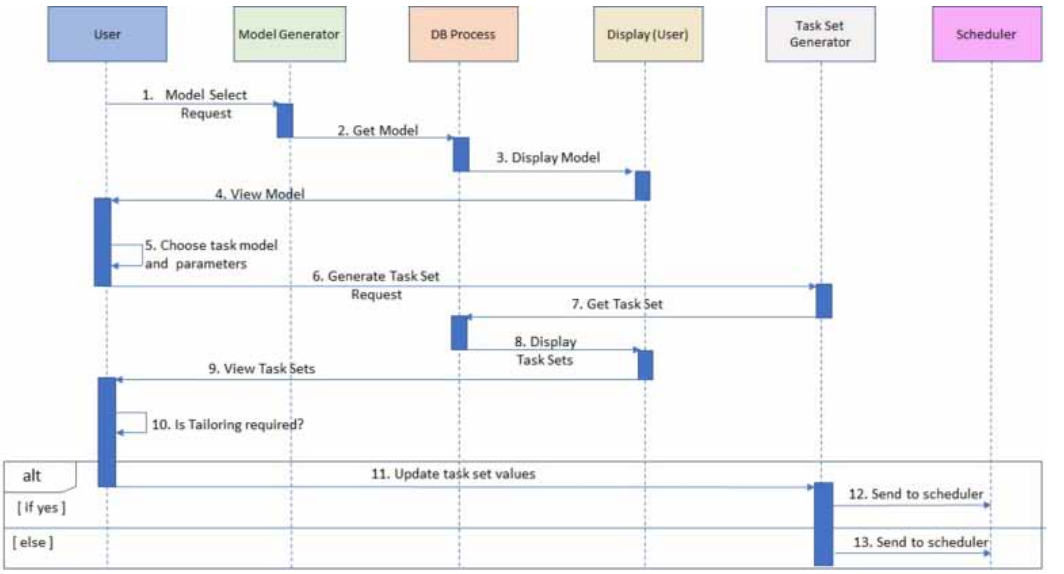


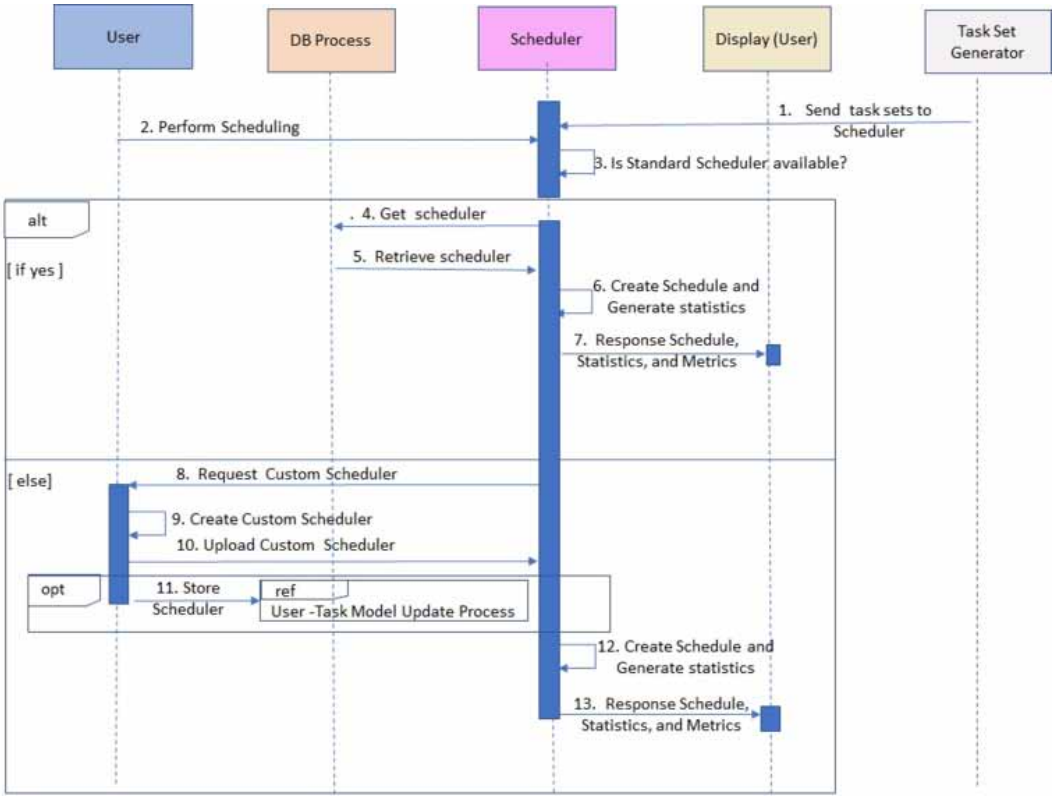
Figure 5. Task model & task set generator sequence



and statistics generated by this scheduler are sent to the user. ARMS provides an option to keep this scheduler in DB with visibility only to that user.

Figure 7 and Figure 8 show the usability of the ARMS and analyze the feasibility of priority ceiling protocol with Earliest-Deadline-First-with-Virtual-Deadlines (EDF-VD) (Zhao et al., 2014) scheduling algorithm for the MCS domain. A user can view task models based on processor configuration and usage scenarios as shown in Figure 7(a) and Figure 8. In the current version, the processor configurations supported are uni-core processor and multi-core processor. The usage scenarios supported are basic,

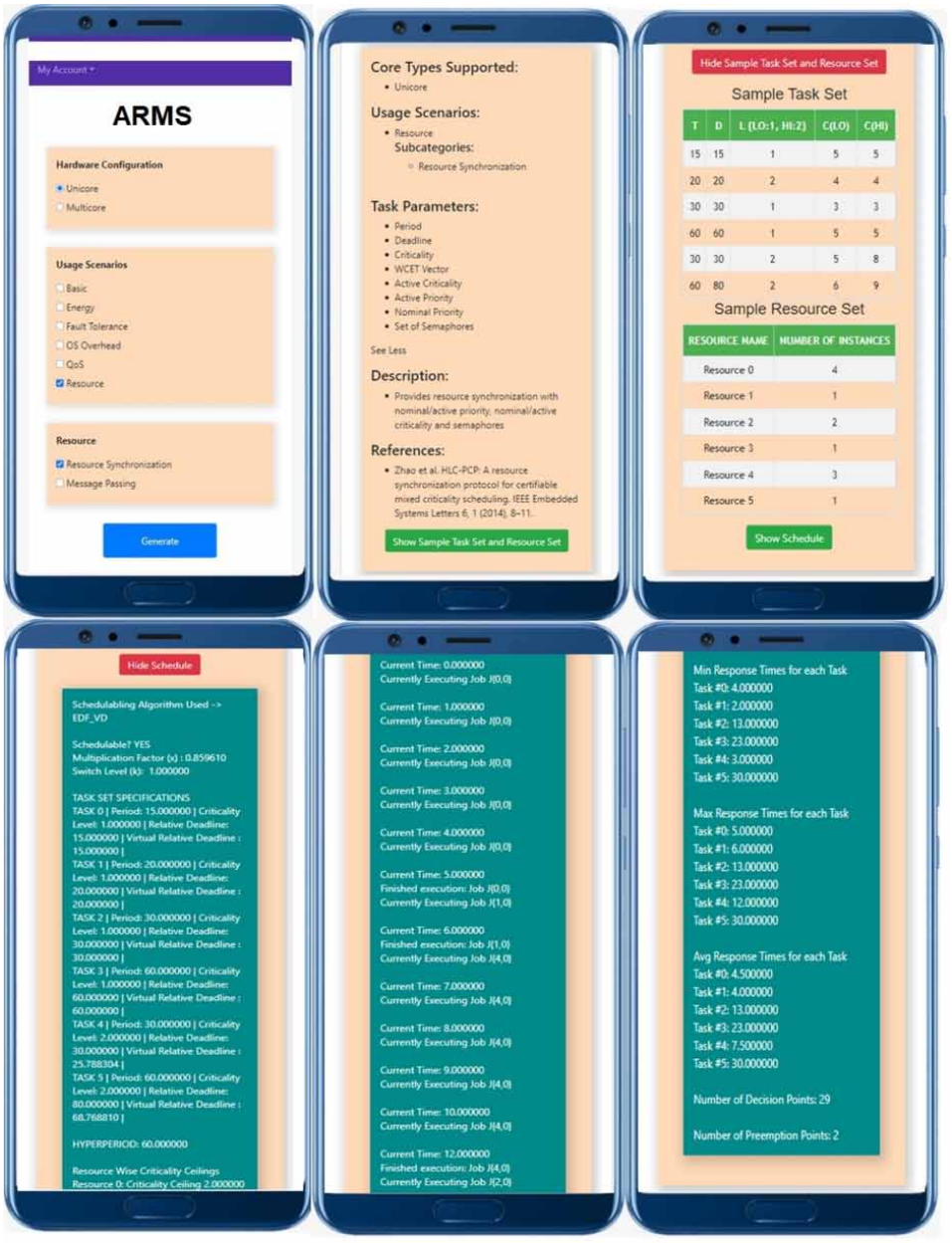
Figure 6. Schedule & statistics generation sequence



resource, QoS, OS overheads, energy and fault tolerance for uni-core and multi-core processor models. Multi-core processor model has parallel processing as an additional usage scenario. There are usage scenarios which require selection of functional attributes to generate task models. For instance, resource usage scenario has resource synchronization, message passing and shared memory as functional attributes. Multiple usage scenarios can also be selected to display task models based on user requirement. The generate option displays the available task models with respect to the chosen attributes with its description and related works in the MCS domain. Figure 7(b) displays the resource-based task models on selection of uni-core processor configuration, resource usage scenario and resource synchronization functional attribute. Based on research requirements, it is possible to choose and customize tasks models to suit the needs of research/usage scenarios. For each of the displayed/customized task models, sample task sets can be generated. As an example, Figure 7(c) displays sample task sets for the chosen resource-based task model with parameters period, deadline, criticality, (WCET) vector, active criticality, active priority, nominal priority and semaphores. Further, the framework assists to choose algorithms for scheduling and profiling. The task sets generated in Figure 7(c) are scheduled using the standard EDF-VD scheduling algorithm as shown in Figure 7(d) The displayed schedule using chosen task sets for a hyper-period is depicted in Figure 7(e)and the Gantt chart of the resultant schedule is shown in Figure 9. The detailed analysis with statistics such as number of preemptions, number of decision points, min/max/average response time of each task are shown in Figure 7(f).

These results explore the usability of ARMS framework for the MCS domain and assist in viewing existing literature, state-of-art research results and choosing the right resource parameters for priority ceiling protocol with resource constraints. It also confirms the usability of priority ceiling with EDF-VD as the protocol for schedulability analysis of resource based MCS.

Figure 7. ARMS - (a) Attributes Selection (b) Attributes Output (c) Task Set (d) Algorithm (e) Schedule (f) Statistics for Resource Task model



Collaboration Phase

The aggregator tool is a freeware web/mobile app that will be made available to users who need to view and decide on appropriate task models for their work. The users will also be able to perform schedulability and comparative analysis. A sustenance team of the working group will play the role of aggregator tool admin. The aggregator platform aims to provide up-to-date information on task models in mixed criticality literature. To facilitate this, the admin performs periodic updates and updates based

Figure 8. ARMS Attributes Selection on web browser

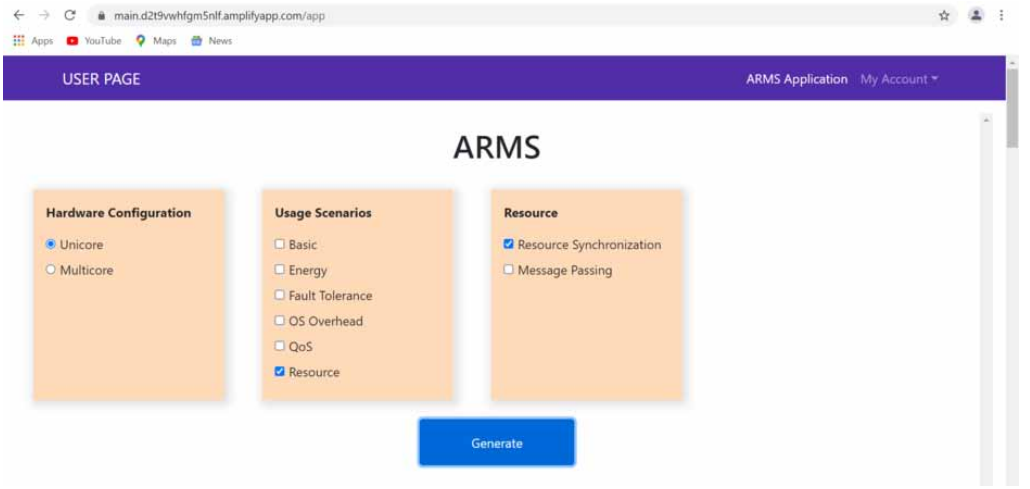
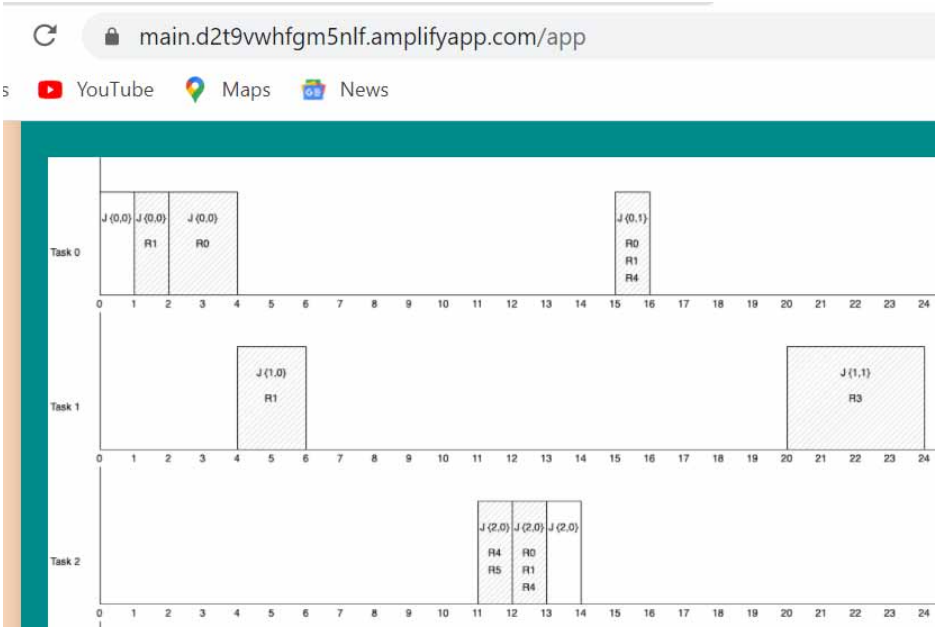


Figure 9. ARMS Schedule on web browser



on request from the user. The swimlane for this process is shown in Figure 10. A user can request for addition of a task model whenever a new task model or new literature is published. The admin reviews the request to confirm its authenticity, parameters and literature. If the model is authentic, an availability check is performed. If the literature corresponding to the model is not present in the DB, new relevant literature is added. If it is a new model, it is added to the DB. The admin performs experimentation and the schedulability analysis results and statistics for the new model is updated in the DB. The admin performs experimentation with the help of author's code base or implements a new code base and gets it validated for correctness from the authors. The schedulability analysis results and statistics for the new model(s) are updated in the DB. The admin also performs task model update periodically based

on literature review conducted by the sustenance consortium. Figure 11 shows the sequence of events when the user sends a request to update/enhance the task model. The user sends a request to admin for updating/ enhancing the task model. The admin process reviews the request and checks its authenticity. It responds with a reject message when the request is not authentic. The admin performs further processing and acknowledges the user accordingly. Figure 12 shows the sequence of events when the admin performs either a period update or an update based on user request. The admin checks if the task model is already available in the DB. If the literature corresponding to the task model is not part of the DB, then the record in the DB is updated with new literature. If task model is not present in the DB, the DB is updated with the new task model along with newly generated statistics and results. If the request originated from a regular user, the Admin process sends an acknowledgement (as shown in Figure 11).

RESULTS AND DISCUSSION

SACRED – Challenges and Best Practices

This section summarizes the challenges faced while developing this software centric collaborative framework. It also proposes the best practices followed to overcome these challenges. The challenges and best practices are listed in Table 2.

Comparative Analysis

The SACRED methodology is compared with few collaborative platforms like HUBZero (Gesing et al., 2019; McLennan & Kennell, 2010), Problem-Based-Programming-Collaborative-Learning-Groupware (PBPCLG) (Chorfi et al., 2022), Open-Innovation-In-Science (OIS) (Beck et al., 2022) and Mentored-Training-for-Dissemination-and-Implementation-Research- in-Cancer (MT-DIRC) (Brownson et al., 2021). HUBZero, PBPCLG, OIS and MT-DIRC focus on collaboration and dissemination of research in the field of engineering, programming, open science and cancer respectively.

HUBZero is a web-based platform that develops simulation and modeling tools through collaboration. Tools hosted on HUBZero use the standard X11 windowing system. Each user is provided their own personal home directory with ownership and required access rights and tools are executed based on specific user's rights. This enables HUBZero to carefully control and track the resource consumption for each tool. HUBZero is deployed using Rappture.

Figure 10. Task model updater and enhancer swimlane

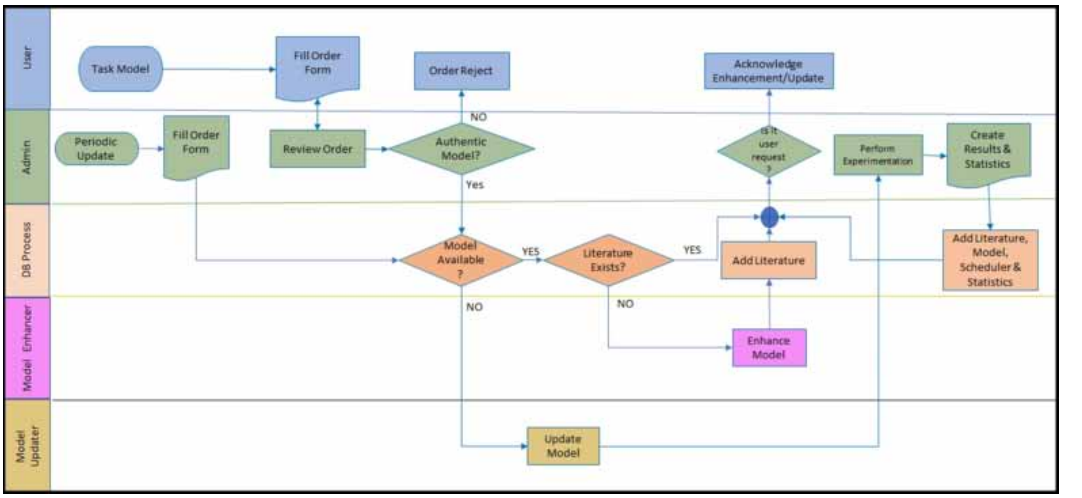


Figure 11. User update sequence

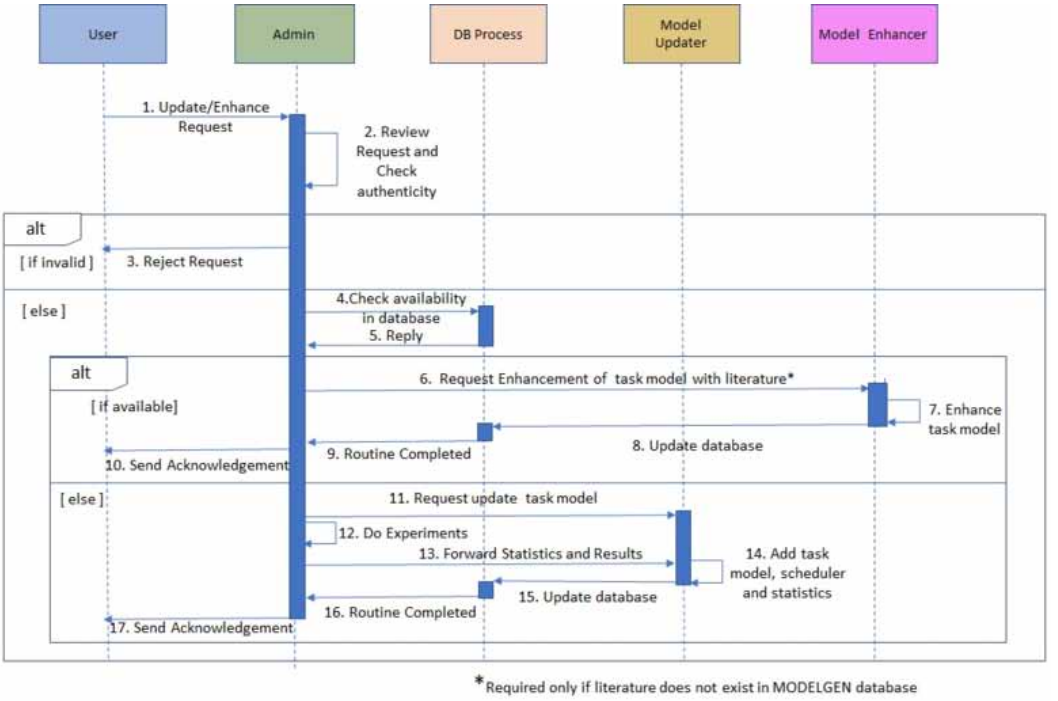


Figure 12. Admin periodic update sequence

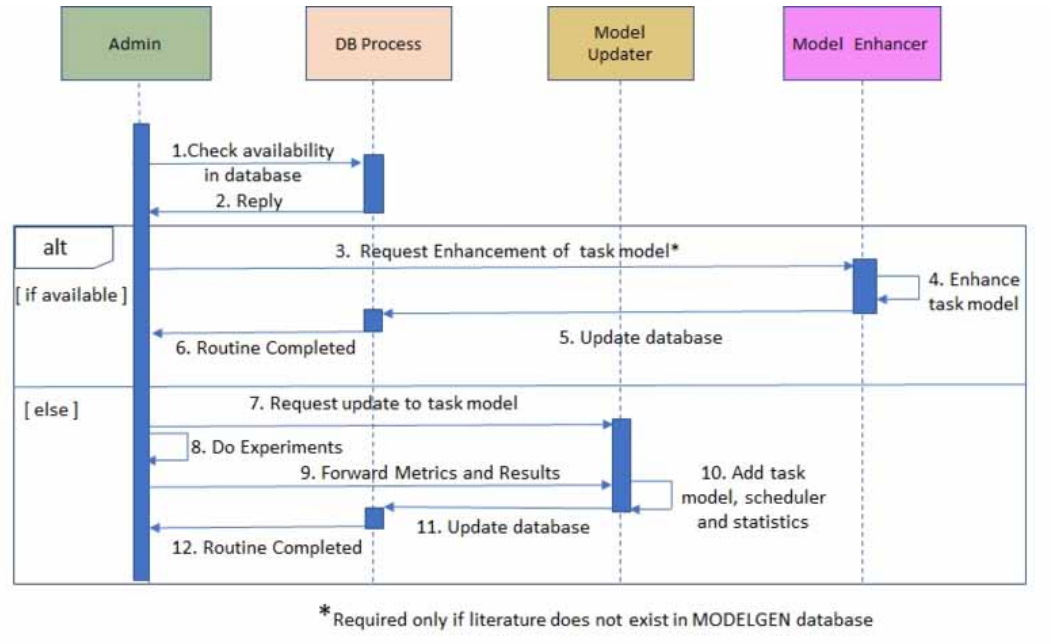


Table 2. SACRED - Challenges and Best Practices (S- Setup, R-Rollout, C- Collaboration)

Sr. No.	Phase	Challenges	Best Practices
1	S, C	Availability of domain expertise	Stakeholder formation, collaborative testing, review by original authors
2	S, R	Proper division of labor among the contributors	Preparation of work instructions by each team mem- ber, detailed work breakdown structure, regular presentations and demos among team members, establishment of the frequency and mode of communication
3	R, C	Proper utilization of available skills	Preparation of work instructions by each team mem- ber, testing by original authors, collaborative testing by peers & like-minded researchers
4	S, R, C	Having common goals	Weekly report submission on progress, template preparation, regular presentations and demos, collaborative testing by peers & like-minded researchers
5	C	Common understanding among stakeholders	Swift and speedy responses among collaborators and their active participation, Up-to-date, relevant and purposeful framework
6	R, C	Uniformity in work product	Preparation of template and guidelines for capturing information
7	S, R, C	Focused plan	Detailed work breakdown structure, Preparation of work instructions by each team member, Weekly report sub- mission on progress and risk assessment
8	C	Regular updates to community without loosing zeal	Up-to-date, relevant and purposeful framework

PBPCLG provides a collaborative platform for problem solving, learning programming and mentoring thus behaving like an educator tech app. It mainly focuses on java programming and problem solving. PBPCLG identifies four types of interactions. The first - individual responsibility, accurately depicts the writer's individual duty towards coding. The second - alternative works is when various developers create distinct variants of the same code snippet and later put them together as one fragment. The third, exchange of dynamic tasks allows developers to share code and ideas. And lastly, collective responsibility allows numerous participants to come to a consensus and create a single code. PBPCLG is deployed using java script.

OIS provides accessibility, clarity, dimension and encourages collaboration and dissemination in the field of science. It aims to transform multidisciplinary knowledge gained through collaboration into innovations through an iterative approach. Experts from several disciplines collaborated to design and develop this framework, emphasizing on the differences and similarities across systems. OIS methodology aspires to influence policy debates and offers direction to researchers and practitioners thereby creating a foundation for ongoing research.

MT-DIRC provides mentored coaching and dissemination of cancer research. Associates were mentored by the MT-DIRC programme, for a period of two-years. They developed distinct innovation and dissemination skills in multiple domains. Annual summer courses were conducted where participants received didactic, small-group, and one-on-one training. Additionally, participants engaged in national conferences and were funded for their research work. Each participant was also allocated a mentor for specialized guidance to achieve the required skill sets. Online and in-person training sessions were also conducted to train mentors.

This work SACRED presents the software framework used in the development of a collaborative platform specifically for the real-time MCS research community. The collaborative platform ARMS, deployed as a cloud-based app/web client provides a research repository and a holistic view to encourage collaboration between industry and academia. It is designed for archiving, updating and reporting existing tasks models in MCS along with their analysis and statistics. ARMS is deployed using ReactJS. Table 3. compares these collaborative frameworks based on the domain of study, purpose, associated tools and deployment.

Table 3. Comparison - Collaborative Frameworks

Features Collab. Frameworks	Domain	Purpose	Associated Tool	Deployment
HUBZero	Engineering	Simulation /Modelling	Web Based	Rappture
OIS	Open Science	Science Research Repository	Not Known	Not Known
MT-DIRC	Medical Science	Cancer Research	Not Known	Not Known
PBPCLG	Programming	Java Language Learning and Mentoring.	Web Based	Java
SACRED	Real Time Systems	MCS Repository, Simulation and Analysis	Cloud Based - ARMS	ReactJs

CONCLUSION AND FUTURE DIRECTIONS

In today's world, it is imperative to have holistic frameworks so that information can be disseminated in order to aid in quick decision making. This work - SACRED proposes a novel approach based on software-centric processes, procedures and practices that can be used to build such collaborative platforms in any domain. The applicability of this approach is demonstrated in a real-life implementation of the collaborative mixed criticality platform ARMS, which serves as a ready-made analyzer for researchers to validate their designs and acts as a quintessential reference aid for academicians and engineers. As future work, it is planned to deploy ARMS in the public domain and deploy it as open source for the research community.

The applicability of SACRED engineering processes, practices and procedures are relevant in a wide range of application domains such as embedded fly-by-wire avionics, robotics, cyber-physical systems, railways, quality education, smart farming and the internet of things. These domains have similar challenges as MCS like lack of tool inter-operability, common interface definitions, uniform parameter nomenclature and adequate parameter syntax/semantics.

COMPETING INTERESTS

The authors of this publication declare there are no competing interests.

FUNDING AGENCY

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. Funding for this research was covered by the author(s) of the article.

REFERENCES

Alahmad, B., Gopalakrishnan, S., Santinelli, L., & Cucu-Grosjean, L. (2011). Probabilities for mixed-criticality problems: Bridging the uncertainty gap. *WiP, RTSS*, 1–4.

Alahmad, B. N., & Gopalakrishnan, S. (2018). Risk-aware scheduling of dual criticality job systems using demand distributions. *Leibniz Transactions on Embedded Systems, 5*(1), 1–1.

Awan, M. A., Bletsas, K., Souto, P. F., Akesson, B., & Tovar, E. (2018). Mixed-criticality scheduling with dynamic memory bandwidth regulation. *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*.

Awan, M. A., Masson, D., & Tovar, E. (2016). Energy efficient mapping of mixed criticality applications on unrelated heterogeneous multicore platforms. *2016 11th IEEE Symposium on Industrial Embedded Systems (SIES)*, 1–10.

Beck, S., Bergenholtz, C., Bogers, M., Brasseur, T.-M., Conradsen, M. L., Di Marco, D., Distel, A. P., Dobusch, L., Dörler, D., Effert, A., Fecher, B., Filiou, D., Frederiksen, L., Gillier, T., Grimpe, C., Gruber, M., Haeussler, C., Heigl, F., Hoisl, K., & Xu, S. M. et al. (2022). The open innovation in science research field: A collaborative conceptualisation approach. *Industry and Innovation*, *29*(2), 136–185. doi:10.1080/13662716.2020.1792274

Borgho, U. M., & Teege, G. (1993). Application of collaborative editing to software-engineering projects. *Software Engineering Notes*, *18*(3).

Brownson, R. C., Jacob, R. R., Carothers, B. J., Chambers, D. A., Colditz, G. A., Emmons, K. M., Haire-Joshu, D., Kerner, J. F., Padek, M., Pfund, C., & Sales, A. (2021). Building the next generation of researchers: Mentored training in dissemination and implementation science. *Academic Medicine*, *96*(1), 86–92. doi:10.1097/ACM.0000000000003750 PMID:32941251

Burns, A. (2013). The application of the original priority ceiling protocol to mixed criticality systems. *Proc. ReTiMiCS, RTCSA*, 7–11.

Burns, A., & Davis, R. I. (2013). Mixed criticality on controller area network. *25th Euromicro Conference on Real-Time Systems*.

Choi, J., Yang, H., & Ha, S. (2018). Optimization of fault-tolerant mixed-criticality multi-core systems with enhanced wcrt analysis. *ACM Transactions on Design Automation of Electronic Systems*, *24*(1), 1–26. doi:10.1145/3275154

Chorfi, A., Hedjazi, D., Aouag, S., & Boubiche, D. (2022). Problem-based collaborative learning groupware to improve computer programming skills. *Behaviour & Information Technology*, *41*(1), 139–158. doi:10.1080/0144929X.2020.1795263

Davis, R. I., Altmeyer, S., & Burns, A. (2018). Mixed criticality systems with varying context switch costs. *Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 140–151. doi:10.1109/RTAS.2018.00024

Dridi, M., Rubini, S., Lallali, M., Flórez, M. J. S., Singhoff, F., & Diguet, J.-P. (2019). Design and multi-abstraction-level evaluation of a noc router for mixed-criticality real-time systems. *ACM Journal on Emerging Technologies in Computing Systems*, *15*(1), 1–37. doi:10.1145/3264818

Ekberg, P., Stigge, M., Guan, N., & Yi, W. (2013). State-based mode switching with applications to mixed criticality systems. *Proc. WMC, RTSS*, 61–66.

Evripidou, C. (2016). *Scheduling for mixed-criticality hypervisor systems in the automotive domain* [Doctoral dissertation]. University of York.

Fleming, T., & Burns, A. (2014). Incorporating the notion of importance into mixed criticality systems. *Proc. 2nd Workshop on Mixed Criticality Systems (WMC), RTSS*, 33–38.

Gesing, S., Zentner, M., Clark, S., Stirm, C., & Haley, B. (2019). Hubzero®: Novel concepts applied to established computing infrastructures to address communities' needs. *Proceedings of the practice and experience in advanced research computing on rise of the machines*. doi:10.1145/3332186.3332238

Giannopoulou, G., Stoimenov, N., Huang, P., & Thiele, L. (2013). Scheduling of mixed- criticality applications on resource-sharing multicore systems. *Proceedings of the Eleventh ACM International Conference on Embedded Software*, 17. doi:10.1109/EMSOFT.2013.6658595

Gill, C., Orr, J., & Harris, S. (2018). Supporting graceful degradation through elasticity in mixed-criticality federated scheduling. *Proc. 6th Workshop on Mixed Criticality Systems (WMC), RTSS*, 19–24.

Gu, X., Easwaran, A., Phan, K.-M., & Shin, I. (2015). Resource efficient isolation mechanisms in mixed-criticality scheduling. *2015 27th Euromicro Conference on Real-Time Systems*, 13–24.

Guo, Z., Santinelli, L., & Yang, K. (2015). Edf schedulability analysis on mixed-criticality systems with permitted failure probability. *2015 IEEE 21st International Conference on Embedded and Real-Time Computing Systems and Applications*, 187–196.

Huang, P., Kumar, P., Giannopoulou, G., & Thiele, L. (2014). Energy efficient dvfs scheduling for mixed-criticality systems. *Proceedings of the 14th International Conference on Embedded Software*, 11. doi:10.1145/2656045.2656057

Huang, P., Kumar, P., Stoimenov, N., & Thiele, L. (2013). Interference constraint graph—a new specification for mixed-criticality systems. *IEEE Conference on Emerging Technologies & Factory Automation (ETFA)*. doi:10.1109/ETFA.2013.6647967

Khan, M. N., Ashraf, M. A., Seinen, D., Khan, K. U., & Laar, R. A. (2021). Social media for knowledge acquisition and dissemination: The impact of the covid-19 pandemic on collaborative learning driven social media adoption. *Frontiers in Psychology*, *12*, 648253. doi:10.3389/fpsyg.2021.648253 PMID:34135814

Lautamäki, J., Nieminen, A., Koskinen, J., Aho, T., Mikkonen, T., & Englund, M. (2012). Cored: Browser-based collaborative real-time editor for java web applications. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, 1307–1316. doi:10.1145/2145204.2145399

Li, Z., & Wang, L. (2016). Memory-aware scheduling for mixed-criticality systems. *International Conference on Computational Science and Its Applications*, 140–156.

Liu, G., Lu, Y., Wang, S., & Gu, Z. (2014). Partitioned multiprocessor scheduling of mixed-criticality parallel jobs. *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, 1–10.

McLennan, M., & Kennell, R. (2010). Hubzero: A platform for dissemination and collaboration in computational science and engineering. *Computing in Science & Engineering*, *12*(2), 48–53. doi:10.1109/MCSE.2010.41

Monnard, K., Benjamins, M. R., Hirschtick, J. L., Castro, M., & Roesch, P. T. (2021). Co-creation of knowledge: A community-based approach to multilevel dissemination of health information. *Health Promotion Practice*, *22*(2), 215–223. doi:10.1177/1524839919865228 PMID:31470741

Nair, A. S., Colaco, L. M., Patil, G., Raveendran, B. K., & Punnekkatt, S. (2019). Mediator-a mixed criticality deadline honored arbiter for multi-core real-time systems. *IEEE/ACM 2 International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*.

Pathan, R. M. (2014). Fault-tolerant and real-time scheduling for mixed-criticality systems. *Real-Time Systems*, *50*(4), 509–547. doi:10.1007/s11241-014-9202-z

Pathan, R. M. (2017). Improving the quality-of-service for scheduling mixed-criticality systems on multiprocessors. *Euromicro Conference on Real-Time Systems*.

Pellizzoni, R., Schranzhofer, A., Chen, J.-J., Caccamo, M., & Thiele, L. (2010). Worst case delay analysis for memory interference in multicore systems. *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*, 741–746.

Peng, X., Babar, M. A., & Ebert, C. (2014). Collaborative software development platforms for crowdsourcing. *IEEE Software*, *31*(2), 30–36. doi:10.1109/MS.2014.31

Stodden, V., Hurlin, C., & Pérignon, C. (2012). Runmycode. org: A novel dissemination and collaboration platform for executing published computational results. *2012 IEEE 8th International Conference on E-Science*, 1–8.

Taherin, A., Salehi, M., & Ejlali, A. (2018). Reliability-aware energy management in mixed-criticality systems. *IEEE Transactions on Sustainable Computing*, *3*(3), 195–208. doi:10.1109/TSUSC.2018.2801123

Tamas–Selicean, D., & Pop, P. (2011). Design optimization of mixed-criticality real-time applications on cost-constrained partitioned architectures. *2011 IEEE 32nd Real-Time Systems Symposium*, 24–33.

Thekkilakattil, A., Dobrin, R., & Punnekkat, S. (2014). Mixed criticality scheduling in fault-tolerant distributed real-time systems. *International Conference on Embedded Systems (ICES)*, 92–97. doi:10.1109/EmbeddedSys.2014.6953097

Vestal, S. (2007). Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. *28th IEEE International Real-Time Systems Symposium (RTSS 2007)*, 239–243. doi:10.1109/RTSS.2007.47

Yun, H., Yao, G., Pellizzoni, R., Caccamo, M., & Sha, L. (2012). Memory access control in multiprocessor for real-time systems with mixed criticality. *2012 24th Euromicro Conference on Real-Time Systems*, 299–308.

Zhao, Q., Gu, Z., & Zeng, H. (2014). Hlc-pcp: A resource synchronization protocol for certifiable mixed criticality scheduling. *IEEE Embedded Systems Letters*, *6*(1), 8–11. doi:10.1109/LES.2013.2273352

Zhao, Q., Gu, Z., & Zeng, H. (2015). Resource synchronization and preemption thresholds within mixed-criticality scheduling. *ACM Transactions on Embedded Computing Systems*, *14*(4), 1–25. doi:10.1145/2783440

Louella Colaco is currently pursuing her PhD at BITS Pilani K. K. Birla Goa Campus. She is also Assistant Professor in the department of Computer Engineering at Padre Conceicao College of Engineering, Verna, Goa. She has 17 years of teaching experience at UG and PG level and has guided several graduate and post graduate dissertations. Her research interest include operating systems, real time systems and mixed criticality systems. Her main areas of research are resource synchronization and energy efficiency in mixed criticality systems.

Arun S. Nair is working as Divisional Manager (Architect at Nexteer Automotive India Software Center and is pursuing Ph.D (Computer Science) at BITS Pilani – Goa campus. He is a postgraduate in Electronics and has 24 years of industry experience in space and automotive electronics. He is the author of research studies published in national/international journals and conference proceedings. His research interest includes mixed-criticality systems, Multicore controller, Automotive functional safety ISO26262, and Automotive applications.

Anurag Madnawat obtained his graduate degree in Computer Science from BITS Pilani K.K. Birla Goa Campus. He works on improving the performance and reliability of virtual machine live migrations. He is interested in the field of operating systems, real-time systems, mixed criticality systems and virtualization. His main area of research is resource synchronization in mixed criticality systems.

Biju K. Raveendran is currently serving as an Associate Professor in the Department of Computer Science and Information Systems, BITS PILANI K. K. BIRLA Goa Campus, Goa, India. He received his PhD from the BITS PILANI, PILANI Campus, Rajasthan in 2009. His research area includes energy efficient multi-core/many-core real-time scheduling, energy efficient memory architecture for multi-core/many-core embedded systems, predictable and dependable real-time/embedded system design, big data systems, etc. He was one of the five recipients of Microsoft Research India Fellowship in 2005 for his PhD work. He is a recipient of Microsoft Young Faculty Award in 2009. He is actively involved in collaborative projects with industries like Microsoft and Aditya Birla Group, etc.