Quality Assessment Document

ACM Articles

---

1. Title: Formal Design and Verification of Self-Adaptive Systems with Decentralized Control

Q1 Problem definition: 2
Clearly states the challenge of designing and assuring correctness of decentralized adaptation logic in self-adaptive systems with multiple interacting MAPE-K loops, and the need for formal methods for such systems (Section 1).

Q2 Problem context: 2
Explicit description of context: modern complex, distributed SASS operating in dynamic environments; MAPE-K feedback loops; decentralized control; interacting adaptation concerns; need for formal design/verification. Includes concrete case studies (smart home, traffic monitoring) to ground the context.

Q3 Research design: 2
Explicitly describes the conceptual and methodological framework: definition of self-adaptive ASMs, formalization of MAPE-K loops, patterns, validation via simulation and scenarios, verification via model checking/metaproperties. Structure of sections clearly outlines the design and analysis approach.

Q4 Contributions: 2
Contributions are explicitly enumerated in the introduction (i–vi), and revisited in Section 8.1: a formalism (self-adaptive ASMs), semantics for MAPE patterns, validation and verification techniques, etc.

Q5 Insights: 2
Provides explicit insights on advantages/shortcomings of the approach, trade-offs vs. other formalisms (Timed Automata, Z, SOTA, etc.), what kinds of assurances can and cannot be provided, scalability issues, and where the approach fits in the broader landscape (Sections 7 and 8).

Q6 Limitations: 2
Has a clear, explicit discussion of limitations: no support for time, uncertainty, non-functional properties, or runtime adaptation; scalability limits of their verification techniques; dependence on ASMs' expressiveness (Sections 7.x and 8.2, plus conclusion).

Total quality score: 12/12

IEEE Articles

---

2. Title: A Learning-Based Framework for Engineering Feature-Oriented Self-Adaptive Software Systems

Q1 Problem definition: 2

The paper clearly states the core problems: concept drift at runtime, dependency/crosscutting of adaptations, and efficiency of runtime analysis. The motivation and research gap are explicit.

Q2 Context : 2

The authors situate the work in architecture-based adaptation and product-line/feature literature, provide a running example system, and describe practical constraints for runtime adaptation.

Q3 Research design:2

The framework design, algorithms, implementation, and empirical evaluation are described in detail. Experimental setup and evaluation methodology are explicit.

Q4 Contributions: 2

Contributions are enumerated and revisited: the FUSION framework, feature-oriented modeling, online learning integration, adaptation planning algorithm, prototype, and empirical results.

Q5 Insights: 2

The paper provides concrete insights about how feature models constrain configuration space, how learning copes with concept drift, and trade-offs between black-box learning and white-box analysis.

Q6 Limitations: 2

Threats to validity and limitations are discussed, including assumptions about feature realizations, dependence on quality of metrics, and evaluation scope.

Total quality score: 12/12

---

3. Title: ADAM: Adaptive Monitoring of Runtime Anomalies in Small Uncrewed Aerial Systems

Q1 Problem definition : 2
Problem and motivation are explicit: resource-constrained sUAS, need for adaptive monitoring, and safety concerns.

Q2 Problem Context: 2
Clear domain context (sUAS, flight controllers, middleware, mission scenarios) and related MAPE-K framing.

Q3 Research design: 2
Describes framework design, knowledge-base construction, activation tree algorithm, implementation, and multi-modal evaluation (logs, simulation, real drones).

Q4 Contributions: 2
Contributions are clearly stated: ADAM framework, activation sequence tree, knowledge base, adaptive monitoring strategies, empirical evaluation.

Q5 Insights: 2
Provides concrete insights about trade-offs (onboard vs offload, canaries vs full detectors), detector activation sequencing, and resource savings.

Q6 Limitations: 1
Threats to validity and some limitations are discussed, but discussion of long-term reuse, cross-platform portability, and industrial deployment barriers is less detailed than other sections.

Total score: 11 / 12

---

4. Title: A Dynamic Software Product Line Approach for Adaptation Planning in Autonomic Computing Systems

Q1 Problem definition: 2
The paper clearly states the problem: how to model adaptation reasoning that integrates context and enables reusable adaptation logic; motivates DSPLs for runtime variability and context-aware planning.

Q2 Problem Context: 2
Strong contextualization in DSPL/feature modeling literature, MAPE-K framing, and related DSPL approaches; concrete use case (Tasklet system) grounds the work.

Q3 Research design: 2
Presents a concrete architecture, formal mapping to SAT, implementation details (FESAS + Sat4J), and an evaluation scenario; methods and workflow are explicit.

Q4 Contributions: 2
Contributions are explicit: (1) adaptation-logic architecture embedding a context feature model (CFM) in the knowledge component; (2) a generic, solver-based planner that can be reused across domains.

Q5 Insights: 2
Provides clear insights about how CFMs reduce reasoning complexity, how priorities/costs resolve conflicts, and trade-offs of SAT-based planning.

Q6 Limitations: 1
Some limitations and assumptions are discussed (e.g., need for developer-supplied rules, mapping to literals), but discussion of scalability limits, industrial deployment barriers, and long-term maintenance is less extensive.

Total score: 11 / 12

---

5. Title: A Self-Healing Technique based on Encapsulated Operation Knowledge

Q1 Problem definition: 2
The paper clearly states the problem: high operational cost and the maintenance burden of operation knowledge for self-healing systems, and the need to improve reusability of operation knowledge.

Q2 Problem Context: 2
Context is explicit: autonomic/self-healing systems, component reuse across organizations, dependency issues, and operational scenarios (server, middleware, DB).

Q3 Research design: 2
Presents a concrete conceptual approach (constraint description + operation description + dependency injection), architecture, prototype implementation, and an example application; methods are described in detail.

Q4 Contributions: 2
Contributions are explicit: encapsulation of operation knowledge at component level, dependency injection mechanism, aspect weaving for non-intrusive integration, and a prototype demonstrating feasibility.

Q5 Insights: 2
Provides clear insights about separation of component-specific vs system-specific knowledge, propagation of constraint satisfaction requests, and maintainability benefits from encapsulation.

Q6 Limitations: 1
Some limitations are discussed (assumptions about command success, need for correct state expressions, reliance on configuration files), but broader empirical evaluation, scalability analysis, and cross-organizational validation are limited**.**

Total score: 11 / 12

---

6. Title: Body Sensor Network: A Self-Adaptive System Exemplar in the Healthcare Domain

Q1 Problem definition: 2
The paper clearly motivates the need for human-centered, safety-critical self-adaptive systems in healthcare and frames the exemplar's goals (reliability vs battery consumption).

Q2 Problem Context: 2
Strong domain context: body sensor networks, ROS implementation, control-theory framing, and three concrete uncertainty scenarios.

Q3 Research design: 2
Explicit artifact design and implementation details (Managing/Managed modules, Knowledge Repository, Simulation), plus reproducible setup and launch file examples.

Q4 Contributions: 2
Contributions are explicit: a reusable exemplar (SA-BSN), scenarios, noise injector, patient profiles, sensor simulations, and an extensible controller implementation.

Q5 Insights : 2
Provides actionable insights on trade-offs between reliability and energy, how control metrics map to adaptation quality, and how to experiment with uncertainties.

Q6 Limitations: 1
The paper documents usage and scenarios but gives limited discussion on large-scale industrial adoption, cross-platform portability beyond ROS, and long-term maintenance of reusable artifacts.

Total score: 11 / 12

---

7. Title: Building Reusable Repertoires for Stochastic Self-Planners

Q1 Problem definition: 2

The paper clearly frames the problem: planners for self-adaptive systems struggle with unanticipated changes and plan reuse is promising but costly; objectives for reusable repertoires are explicit.

Q2 Problem Context: 2

Strong contextualization in MAPE-K planning, genetic programming planners, and cloud exemplar inspired by AWS; change/uncertainty model is well described.

Q3 Research design: 2

Presents a two-phase offline/online approach, concrete algorithms (chaos-inspired scenario generation, clone detection, rule-based transforms), implementation choices (Deckard, Comby), and an empirical evaluation on a cloud exemplar.

Q4 Contributions: 2

Contributions are enumerated: chaos-style scenario generation, clone-based extraction, syntactic transformation rules, and empirical results showing improved replanning effectiveness.

Q5 Insights: 2

Provides clear insights on tradeoffs (timeliness vs optimality), why certain plan fragments generalize, and how repertoire construction affects stochastic search.

Q6 Limitations: 1

Some limitations are acknowledged (tradeoffs, evaluation scope), but discussion of large-scale industrial deployment, long-term maintenance of repertoires, and cross-domain generality is limited.

Total score: 11 / 12

---

8. Title: DARE: A Distributed Adaptation and Failure Recovery Framework for Software Systems

Q1 Problem definition: 2
The paper clearly states the problem: providing decentralized self-configuration and self-healing for large dynamic component-based systems and integrating architecture discovery with recovery/adaptation connectors.

Q2 Problem Context: 2
The target context (component-based software architectures, assumptions about failures, cluster experiments with an Emergency Monitoring System exemplar) is explicit.

Q3 Research design: 1
Design is described (architecture, state machines, integration of DeSARM and RAC), and experiments are reported, but experimental design details (metrics, statistical analysis, threats to validity) are limited.

Q4 Contributions : 2
Contributions are explicit: DARE architecture, integration approach, and experimental validation demonstrating decentralized recovery and adaptation.

Q5 Insights: 1
The paper reports empirical recovery times and traces and discusses scalability qualitatively; deeper analysis of tradeoffs and limitations is limited.

Q6 Limitations: 1
Some assumptions and constraints are stated (fail-stop model, reliable transport, synchronized clocks), but broader limitations and generalizability discussion are brief.

Total Quality Score: 9 / 12

---

9. Title: Dynamic QoS Management and Optimization in Service-Based Systems (QoSMOS)

Q1 Problem definition: 2
The paper clearly states the problem: achieving and maintaining QoS in dynamic service-based systems and the need for adaptive, predictable QoS management.

Q2 Problem Context: 2
Strong context: service-based systems, QoS dimensions (performance, reliability, cost), related work survey, and a concrete TeleAssistance case study.

Q3 Research design: 2
Explicit architecture and toolchain (ProProST, PRISM, KAMI, GPAC), description of mechanisms (service selection, resource allocation), and validation via simulation and case study.

Q4 Contributions: 2
Clear contributions: formal QoS specification with probabilistic temporal logics, model-based QoS evaluation using probabilistic model checking, runtime parameter learning, and an integrated adaptation framework (QoSMOS).

Q5 Insights: 1
Provides useful conceptual and methodological insights (value of formal specs, model learning, tradeoffs), but practical evidence about runtime overheads, developer effort, and industrial applicability is limited.

Q6 Limitations: 1
The paper acknowledges validation via simulation and a case study but gives limited discussion on engineering costs (model construction, runtime model-checking overhead), toolchain integration effort, and developer usability.

Total score: 10 / 12

---

10. Title: Dynamic Self-Adaptation for Distributed Service-Oriented Transactions

Q1 Problem definition: 2
The paper clearly states the problem: how to realize reusable, correct dynamic adaptation for service-oriented systems that coordinate distributed (stateful) transactions.

Q2 Problem Context: 2
Strong context: SOA, transaction coordination (Two-Phase Commit), MAPE-style reference architecture, and the SASSY framework are explicitly described.

Q3 Research design: 2
Design is explicit: definition of adaptation patterns, state-machine models, connector abstractions, and a prototype validation (Emergency Response System). Methods (sequence diagrams, state machines, connector encapsulation) are described and applied.

Q4 Contributions: 2
Contributions are explicit: a transaction-aware adaptation pattern, connector/state-machine encapsulation, adaptation sequencing rules (passivate/reactivate), and a SASSY-based prototype.

Q5 Insights: 2
The paper gives actionable insights about quiescence, buffering/queueing strategies, constraints on when components can be adapted, and how connectors mediate adaptation safely.

Q6 Limitations: 1
Limitations are discussed implicitly (assumptions about coordination protocols, focus on Two-Phase Commit, limited empirical evaluation), but there is limited explicit discussion of scalability, heterogeneity of real-world services, or developer effort required to adopt patterns.

Total score: 11 / 12

---

11. Title: Efficient Utility-Driven Self-Healing Employing Adaptation Rules for Large Dynamic

Q1 Problem definition: 2
The paper clearly states the problem: trade-offs between scalable rule-based adaptation and optimal utility-driven planning for architecture-level self-healing, and the goal of combining both to get optimal yet scalable decisions.

Q2 Problem Context: 2
Context is explicit: architecture-based self-adaptation, runtime models, MAPE-K loops, and the mRUBIS benchmark (a modular RUBiS variant) used for validation.

Q3 Research design: 1
Design is described (pattern-based utility, rule definitions, incremental scheme, experiments vs. baselines). However, some experimental design choices (parameter sensitivity, threat to validity discussion) are less detailed than ideal.

Q4 Contributions: 2
Contributions are explicit: pattern-based utility definition, linking utility to pattern-based adaptation rules, an incremental greedy scheme that guarantees optimality under stated assumptions, and empirical evaluation showing scalability.

Q5 Insights: 2
Provides concrete insights: how pattern coupling enables incremental utility computation, how rule ordering by impact maximizes reward, and tradeoffs vs. constraint-solver approaches.

Q6 Limitations: 1
The paper acknowledges assumptions and some scalability claims, but discussion of practical engineering costs (rule authoring, utility elicitation, cross-domain reuse) is limited.

Total score: 10 / 12

---

12. Title: Evolving an Adaptive Industrial Software System to Use Architecture-Based Self-Adaptation

Q1 Problem definition: 2
The paper clearly states two concrete research questions: whether architecture-based self-adaptation can be applied to legacy systems that already contain built-in adaptation, and what the engineering effort and trade-offs are. The motivation (industrial middleware, DCAS) is explicit.

Q2 Problem Context: 2
Strong contextualization: an industrial middleware (DCAS) used in energy systems, description of DCAS architecture, existing adaptation mechanisms, and the Rainbow framework background. The target domain and constraints are well described.

Q3 Research design: 2
The study is an engineering case study: they evolve DCAS, implement a translation layer, customize Rainbow (models, operators, tactics, strategies), and evaluate effort, performance, and maintainability. The steps and artifacts are described in sufficient detail to reproduce the integration approach.

Q4 Contributions: 2
Contributions are explicit: a concrete integration of Rainbow with an industrial system, a description of the translation infrastructure, customization points, measured engineering effort and performance observations, and lessons learned about refactoring and adoption.

Q5 Insights: 2
The paper provides actionable insights (refactoring cost, instrumentation needs, performance trade-offs, how to map legacy controls to architecture-based operators/tactics, and effort estimates from prior Rainbow case studies for comparison).

Q6 Limitations: 1
The paper discusses practical limits (manual scale-out, need to expose internals, domain specificity), but the discussion of external validity/generalizability is limited : the lessons are strong for similar middleware but less explicit about how they transfer to very different legacy systems.

Total score: 11 / 12

---

13. Title: FESAS IDE: An Integrated Development Environment for Autonomic Computing

Q1 Problem definition: 2
The paper clearly motivates the complexity of developing adaptation logic (AL) for self-adaptive systems and argues that finer-grained reuse and tool support can reduce that complexity.

Q2 Problem Context: 2
Strong contextualization: MAPE-K background, related frameworks (Rainbow, SASSY), and a cloud/VM example scenario are provided to ground the work.

Q3 Research design: 1
The paper describes the IDE, its components, and a five-case evaluation. However, the evaluation is exploratory (example cases) rather than controlled or quantitative, so the research design is only moderately rigorous.

Q4 Contributions: 2
Contributions are explicit: (1) FESAS IDE (Development and Design tools), (2) a development/deployment process, (3) a repository and component template for fine-grained reuse.

Q5 Insights: 2
Provides actionable insights about separation of concerns in AL components, benefits of contract/metadata driven reuse, and practical developer workflows (packaging, testing, deployment).

Q6 Limitations: 1
The paper acknowledges limitations (tooling tied to Eclipse/EMF/GMF, limited empirical validation, runtime AL adaptation out of scope) but lacks deep quantitative evaluation or large-scale case studies.

Total score: 10 / 12

---

14. Title: Flexible Self-organisation for the Cloud-Edge Continuum: a Macro-programming Approach

Q1 Problem definition: 2
The paper clearly states the problem: how to modularize macro-programs (field/aggregate programs) so parts can be deployed across cloud/edge/device tiers while preserving collective/self-stabilising behaviour and gaining non-functional benefits.

Q2 Context : 2
Strong context: aggregate/macro-programming, edge-cloud continuum, collective services (gradients, steering, emergency detection), and a realistic rescue/e-health case study ground the work.

Q3 Research design : 2
Well-structured research design: formal model (operational semantics), proof sketch of

deployment-independence (self-stabilisation preservation), implementation, and simulation experiments to evaluate correctness and non-functional trade-offs.

Q4 Contributions: 2
Contributions are explicit: (1) a modularisation framework for macro-programs, (2) a forwarding/offloading mechanism for multi-tier deployment, (3) formal semantics and deployment-independence result, (4) implementation + simulation and open-source artefact.

Q5 Insights: 2
Provides concrete insights: how to partition macro-programs into local vs collective components, how forwarding chains preserve semantics, and trade-offs (energy, deployability) demonstrated in simulation.

Q6 Limitations: 1
Limitations are acknowledged implicitly (assumptions on network model, reliance on simulation, heterogeneity handling) but the paper gives limited empirical validation on real hardware or large heterogeneous deployments.

Total score: 11 / 12

---

15. Title: Improving Context-Awareness in Self-Adaptation using the DYNAMICO Reference Model

Q1 Problem definition: 2
The paper clearly states the problem: monitoring infrastructures and adaptation goals change at runtime, undermining self-adaptation unless monitoring itself is adaptive. The motivation and scope are explicit.

Q2 Context : 2
Strong contextual grounding: Znn.com exemplar, SOA governance scenario, SLAs and renegotiation, and concrete mapping to monitoring and adaptation concerns.

Q3 Research design : 1
Implementation-driven: the paper presents an implemented realization (SMARTERCONTEXT + QOS-CARE) and an experimental comparison with Rainbow/Znn.com. However, experimental design details (metrics, statistical treatment, repeatability) are described but not as rigorously as in controlled empirical studies.

Q4 Contributions : 2
Contributions are explicit: (1) DYNAMICO reference model (three feedback loops), (2) an implementation integrating SMARTERCONTEXT and QOS-CARE, and (3) comparative evaluation vs. Rainbow/Znn.com.

Q5 Insights : 1
Provides useful insights about the benefits of adaptive monitoring and the feasibility of synthesizing monitoring strategies from SLA specifications, but some claims are qualitative and evaluation coverage is limited.

Q6 Limitations : 1
The paper acknowledges limits (case-study scope, reliance on specific middleware, and limited generalization), but discussion of reuse-specific limitations (e.g., artifact packaging, API stability) is limited.

Total score: 9 / 12

---

16. Title: Language Support for Modular Autonomic Managers in Reconfigurable Software Components

Q1 Problem definition : 2
The paper clearly states the problem: lack of modularity in languages and toolchains for specifying autonomic managers prevents reuse, substitutability and scalable controller synthesis.

Q2 Context : 2
Strong, concrete context: component-based systems, MAPE/autonomic managers, discrete controller synthesis (DCS), Heptagon/BZR intermediate language, and a realistic case study (RUBIS/Brownout) and runtime (FraSCAti).

Q3 Research design : 2
Explicit engineering approach: language extension (Ctrl-F), translation to Heptagon/BZR, modular DCS, prototype implementation, and experimental comparison (modular vs monolithic compilation).

Q4 Contributions : 2
Contributions are explicit: modularity extensions to Ctrl-F (inheritance, overriding), contract-based abstractions, automated translation to modular Heptagon/BZR, and modular DCS to tame combinatorial explosion.

Q5 Insights : 2
Provides actionable insights: how modular contracts enable reuse and substitutability, how modular compilation reduces synthesis cost, and trade-offs between abstraction and visibility for verification.

Q6 Limitations : 2
The paper explicitly discusses limits: dependence on DCS (inherent complexity), assumptions required for modular contracts, and the need for toolchain support (Heptagon/BZR, FraSCAti) which constrains portability.

Total score: 12 / 12

---

17. Title: Learning Recovery Strategies for Dynamic Self-healing in Reactive Systems

Q1 Problem definition : 2
The paper clearly motivates the limits of conventional self-healing (predefined hooks, coarse-grained fixes, inability to handle unanticipated fine-grained failures) and states the goal: enable run-time learning of recovery strategies for reactive systems.

Q2 Problem Context : 2
Strong, concrete context: reactive systems, REScala/ContextScala/COP, two case studies (mouse-tracking prototype and DeltaIoT exemplar). Domain and constraints are explicit.

Q3 Research design : 2
Explicit artifact design (monitor architecture, RL learning model, variation manager), prototype implementation, and empirical validation on two case studies with measurable outcomes.

Q4 Contributions : 2
Contributions are clearly enumerated: flexible declarative monitors, RL-based recovery learning, extraction of learned strategies as COP variations, and empirical evaluation.

Q5 Insights : 2
Provides actionable insights about monitor expressiveness, trade-offs between exploration and coverage, feasibility of packaging learned sequences as context variations, and applicability to reactive and IoT exemplars.

Q6 Limitations : 1
The paper acknowledges limitations (dependence on monitor expressiveness and exploration coverage, potential scalability and generalization issues) but discussion of large-scale deployment, cross-platform portability, and long-term maintenance of learned artifacts is limited.

Total score: 11 / 12

---

18. Title: Lifelong Self-Adaptation: Self-Adaptation Meets Lifelong Machine Learning

Q1 Problem definition: 2
The paper clearly states the problem: ML models used in self-adaptive systems fail when new learning tasks emerge (e.g., concept drift), motivating a lifelong-learning layer to evolve models at runtime.

Q2 Problem Context: 2
Strong context: architecture-based self-adaptation, MAPE-K managing systems, concrete focus on concept drift, and two exemplar case studies (DeltaIoT and a gas delivery case).

Q3 Research design: 1
The paper presents a reusable architecture and two concrete instantiations, plus empirical validation on two case studies. However, experimental design details (e.g., statistical rigor, parameter sensitivity, repeatability settings) are described but not exhaustively controlled, so some reproducibility details are limited.

Q4 Contributions: 2
Contributions are explicit: (1) a reusable architecture for lifelong self-adaptation, (2) two instantiations for sudden and incremental concept drift, (3) validation on two case studies.

Q5 Insights: 2
Provides actionable insights about how a lifelong ML loop can detect new tasks, collect task-specific knowledge, and evolve learning models; discusses trade-offs (trigger frequency, knowledge management, learner choices).

Q6 Limitations: 1
The paper acknowledges limits (focus on learning-model evolution only, not runtime software evolution; domain-specific choices; open issues like catastrophic forgetting and under-specification) but discussion of operational costs, scalability of knowledge repositories, and human-in-the-loop burdens is brief.

Total score: 10 / 12

---

19. Title: MOSES: A Framework for QoS Driven Runtime Adaptation of Service-Oriented Systems

Q1 Problem definition: 2
The paper clearly motivates runtime QoS-driven adaptation for service-oriented systems, frames the problem space, and states why existing per-request selection approaches are insufficient.

Q2 Problem Context: 2
Strong domain context: SOA, composite services, SLAs, flows of requests, and concrete examples (workflow patterns, DeltaIoT-style scenarios) are used to ground the work.

Q3 Research design: 2
Explicit methodology: formal problem-space characterization, model definitions, LP optimization formulation, prototype implementation, and extensive experimental evaluation.

Q4 Contributions: 2
Contributions are explicit: problem-space taxonomy, MOSES methodology and tool, SLA and QoS models, LP-based planner integrating service selection and coordination patterns, and experimental validation.

Q5 Insights: 2
Provides actionable insights on trade-offs (service selection vs coordination patterns), when coordination patterns are necessary, modeling requirements for QoS prediction, and computational costs of optimization.

Q6 Limitations: 2

Paper documents limitations and assumptions: scope restricted to certain workflow patterns, single-authority control, reliance on accurate QoS/service descriptions and monitoring, and computational scalability concerns; these are discussed and evaluated.

Total score: 12 / 12

---

20. Title: Managing Uncertainty in Self-Adaptive Systems with Plan Reuse and Stochastic Search.

Q1 Problem definition: 2

The paper clearly states the problem: how to handle post-design-time uncertainty in self-adaptive systems by reusing prior planning knowledge instead of replanning from scratch. Motivating examples and concrete change types are given (tactic cost/effect changes, new tactics, environment/use-case shifts).

Q2 Problem Context: 2

Good situating in MAPE-K planning literature, prior online/offline planners, PRISM/MDP approaches, and prior uses of stochastic search and plan reuse in AI/program repair.

Q3 Research design: 2

Presents a concrete GP-based planner, formal plan representation, fitness evaluation via simulation (state trees), concrete reuse techniques (scratch_ratio, kill_ratio, trimmings), and an empirical evaluation (benchmarks, PRISM comparison, statistical testing).

Q4 Contributions: 2

Contributions are explicit: (1) empirical finding that naïve plan reuse can reduce utility; (2) a set of techniques to make reuse effective; (3) a GP planner implementation and comparison to exhaustive PRISM MDP planning; (4) empirical analysis of time, quality, and diversity trade-offs.

Q5 Insights: 2

Provides actionable insights (why naïve reuse fails, how to reduce reuse cost, how starting plan objectives bias multi-objective search) and novel engineering techniques to enable reuse in GP planning for self-adaptive systems.

Q6 Limitations: 1

The paper acknowledges limitations (e.g., model synchronization out of scope, hardware/parallelism assumptions, domain specificity) but discussion of large-scale industrial deployment and long-term plan library maintenance is limited.

Total score: 11 / 12

---

21. Title: Proactive Adaptation of Service Composition

Q1 Problem definition: 2
The paper clearly states the problem: supporting proactive adaptation of service compositions (C1–C4) and limitations of prior work (reactive only, limited matching) in the introduction.

Q2 Problem Context: 2
Context is explicit: service-based systems, SLAs, BPEL compositions, QoS concerns, runtime execution models and monitoring; example business process used to ground the approach.

Q3 Research design: 1
The paper describes architecture, components (composer, execution engine, adaptor, monitor, discovery), and a prototype; however, experimental design and evaluation details are limited and some techniques (e.g., service discovery) are assumed or referenced rather than fully described.

Q4 Contributions: 2
Contributions are enumerated (support for C1–C4, 1-1/1-n/n-1/n-m replacements, signature dependency handling, prototype).

Q5 Insights: 1
The paper provides practical insights (preference for signature matches, use of grouping, dependency analysis) but many claims are described at a conceptual level without deep empirical evidence.

Q6 Limitations: 1
Some limitations are acknowledged (focus on stateless services, simple monitoring, limited QoS prediction detail), but the discussion is not exhaustive.

Total score: 9 / 12

---

22. Title: RAMSES: an Artifact Exemplar for Engineering Self-Adaptive Microservice Applications.

Q1 Problem definition: 2
The paper explicitly states the problem: enabling reusable, decoupled self-adaptation for microservice applications and easing reuse of adaptation logic via a Contract-First, API-led autonomic manager.

Q2 Problem Context: 2
Context is explicit: microservice architectures, MAPE-K control loops, QoS concerns (availability, response time), and limitations of built-in infra features (autoscaling, circuit breakers).

Q3 Research design: 1
The paper describes the exemplar architecture, implementation choices (Spring Boot/Cloud), APIs, and a

ScenarioRunner testbed; experimental setup and metrics are presented but evaluation detail is preliminary (limited scenarios, short observation windows).

Q4 Contributions: 2
Contributions are explicit: a non-simulated exemplar, a reusable autonomic manager with RESTful probing/actuating APIs, an e-food managed application, and publicly available artifacts.

Q5 Insights: 1
Practical insights are provided (API-led reuse, separation of concerns, adaptation strategies), but deeper empirical analysis and generalization claims are limited.

Q6 Limitations: 1
The paper acknowledges limits (focus on stateless microservices, preliminary experiments, reliance on specific infra choices) but does not exhaustively explore scalability or cross-platform integration issues.

Total quality score: 9 / 12

---

23. Title: REACT: A Model-Based Runtime Environment for Adapting Communication Systems

Q1 Problem definition: 2
The paper explicitly states the problem: enabling domain experts to add self-adaptivity to heterogeneous, distributed communication systems with low development effort.

Q2 Problem Context: 2
Context is clear: communication systems (IoT, SDN, cloud), heterogeneity, decentralization, legacy integration, and practitioner needs are described.

Q3 Research design: 1
The paper presents architecture, implementation, a development process, and two case evaluations (cloud resource management and SDN). Experimental design and metrics are described but some evaluation details are summarized rather than exhaustively specified.

Q4 Contributions: 2
Contributions are explicit: a reusable runtime environment (REACT), model-based specification using Clafer/UML, language-independent sensor/effector interfaces, decentralized deployment support, open-source prototype, and comparative evaluation with Rainbow.

Q5 Insights:
The paper provides explicit insights about developer barriers (heterogeneity, decentralization, tooling), the value of separating problem/solution models, and practical trade-offs for practitioner adoption.

Q6 Limitations: 1
Limitations are acknowledged (need for domain expert modeling, reliance on provided models, evaluation scope), but some scalability and long-term reuse concerns are not deeply empirically explored.

Total score: 10 / 12

---

24. Title: ReBeT: Architecture-based Self-adaptation of Robotic Systems through Behavior Trees

Q1 Problem Definition: 2

The paper provides an explicit problem definition, stating that current self-adaptive robotic systems use ad-hoc, non-reusable mechanisms for architectural adaptation and lack explicit consideration of quality requirements.

Q2 Problem Context: 2

The context is explicitly described, covering the state of robotics software (ROS2, Behavior Trees), the gap in quality-driven and reusable architectural adaptation, and related work.

Q3 Research Design: 2

The design of the ReBeT approach (QRDecorators, AAL, AdaptDecorators) and the evaluation methodology (experiments with the FROG robot) are explicitly described.

Q4 Contributions: 2

Three clear contributions are listed: 1) QRDecorators, 2) Architecture Adaptation Layer (AAL), 3) Adaptation Decorators (AdaptDecorators).

Q5 Insights: 2

Explicit insights are provided from the quantitative evaluation (utility increase) and a discussion on the added value of the abstractions for development.

Q6 Limitations: 2

A dedicated "Threats to Validity" section and a discussion on limitations of integrating with existing ROS2 systems are provided.

Total Score: 12 / 12

---

25. Title: Reusable Self-Adaptation through Bidirectional Programming

Q1 Problem Definition: 2
Explicitly defines the problem: reusing adaptation strategies requires abstract models, but correctly synchronizing concrete configuration files with these abstract models is non-trivial and error-prone.

Q2 Problem Context: 2
Explicitly describes the context of self-adaptive systems (MAPE-K loop), the separation between target system and adaptation layer, and the state of reuse in adaptation frameworks.

Q3 Research Design: 2
Explicitly describes the proposed approach (bidirectional transformations/BiGUL), the design of the synchronization mechanism, and the case study setup with two scenarios (adaptation and migration).

Q4 Contributions: 2
Clearly states the contribution: a synchronization mechanism correct by construction using putback-based bidirectional programming to enable reusable adaptation layers.

Q5 Insights: 2
Provides explicit insights from the case study, demonstrating successful adaptation and migration, and discusses the handling of configuration file features (defaults, context overriding).

Q6 Limitations: 2
Includes a dedicated "Threats to Validity" section discussing internal and external validity (e.g., simulated MAPE loop, assumption of no concurrent modifications).

Total Score: 12 / 12

---

26. Title: SUAVE: An Exemplar for Self-Adaptive Underwater Vehicles

Q1 Problem Definition: 2
Explicitly defines the problem: AUVs operate in uncertain environments without human supervision, requiring self-adaptation. There is a need for reusable exemplars to facilitate research.

Q2 Problem Context: 2
Explicitly describes the context: underwater robotics, pipeline inspection missions, specific uncertainties (thruster failure, visibility), and the use of ROS 2 and Metacontrol.

Q3 Research Design: 2
Explicitly describes the design of the two-layered exemplar, the managed/managing subsystem split, the integration with Metacontrol, and the experimental evaluation.

Q4 Contributions: 2
Clearly states two contributions: 1) a reusable ROS 2 exemplar for self-adaptive AUVs, and 2) a reusable Metacontrol-based adaptation logic baseline.

Q5 Insights: 2
Provides explicit insights from the evaluation, showing performance improvement with Metacontrol and discussing the exemplar's extensibility.

Q6 Limitations: 1
Discusses simplifications of the use case and lists future extensions, but lacks a dedicated "Threats to Validity" or "Limitations" section that explicitly critiques the approach.

Total Score: 11 / 12

---

27. Title: Self-Adaptive Manufacturing with Digital Twins

Q1 Problem Definition: 2
Explicitly defines the problem: long-living Cyber-Physical Production Systems (CPPSs) diverge from intended behavior, requiring human expertise for adaptation. The challenge is to capture and reuse this expertise for autonomous self-adaptation.

Q2 Problem Context: 2
Explicitly describes the context of Industry 4.0, CPPSs, Digital Twins (DTs), and the specific domain of injection molding. Requirements (R1–R4) are clearly stated to frame the solution.

Q3 Research Design: 2
Explicitly describes the design of a modeling framework with multiple Domain-Specific Languages (DSLs), a modular DT architecture integrating Case-Based Reasoning (CBR), and an application to an injection molding machine with performance evaluation.

Q4 Contributions: 2
Clearly states two contributions: 1) extensible modeling languages for capturing domain expertise (cases and similarity), and 2) a modular architecture for integrating CBR into DTs.

Q5 Insights: 2
Provides explicit insights from the application, including measured cycle times for CBR, discussion of language usability for domain experts, and the framework's reusability for other CPPSs.

Q6 Limitations: 1
Discusses challenges and limitations (e.g., domain experts' difficulty in explicitly expressing similarity, safety concerns with autonomous adaptation) but lacks a dedicated "Threats to Validity" section with a structured critique.

Total Score: 11 / 12

---

28. Title: Self-Management of Adaptable Component-Based Applications

Q1 Problem Definition: 2
Explicit problem definition provided in the Abstract and Introduction: how to automatically select appropriate individual component adaptations to address deviations from desired system behavior.

Q2 Problem Context: 2
Explicit context description: software systems built from adaptable components with multiple implementations/configuration options, focusing on self-optimization and adaptation.

Q3 Research Design: 2
Explicit description of the two-phase (offline/online) approach, rule generation, evaluation, and experimental case study setup.

Q4 Contributions: 2
Explicit contributions listed in the Abstract and Conclusion: a general framework for automatic adaptation decision-making, leveraging component developer knowledge, with offline rule generation and online evaluation.

Q5 Insights: 2
Explicit insights from experience discussed in Section 8.4, covering flexibility, extensibility, and applicability constraints.

Q6 Limitations: 1
General limitations described in Section 8.4 (e.g., constraints on dynamic component creation, challenges in impact estimation). No dedicated "Limitations" section but discussed within the text.

Total Score: 11 / 12

---

29. Title: Self-Adaptive Containers: Building Resource-Efficient Applications with Low Programmer Overhead

Q1 Problem Definition: 2
Explicit problem definition in Abstract and Introduction: developing scalable, resource-efficient applications that adapt to environmental constraints with low programmer effort, avoiding manual refactoring for each new environment.

Q2 Problem Context: 1
General context description: software must run in diverse environments (servers, mobile) with different resource constraints and SLOs. The need for self-adaptation and reuse across contexts is stated but not deeply elaborated.

Q3 Research Design: 2
Explicit description of the library's architecture (API, Self-adaptive Unit with Observer, Analyzer, Adaptor), design decisions, and experimental case study setup.

Q4 Contributions: 2
Three explicit contributions listed at the end of Section I: a resource-aware container library, use of probabilistic/out-of-core techniques, and a way to easily adapt naive algorithms.

Q5 Insights: 1
General insights from the case study are presented (performance gains, influence of SLO priority), but a dedicated "Insights" section or explicit discussion of broader implications is not provided.

Q6 Limitations: 1
General limitations described in Future Work (Section VI): library functionalities are not yet complete, and programmer overhead could be further reduced via automated code analysis tools.

Total Score: 9 / 12

---

30. Title: Supporting Self-Adaptation via Quantitative Verification and Sensitivity Analysis at Run Time

Q1 Problem Definition: 2
The paper explicitly defines the problem of traditional model checking being too computationally expensive for runtime use in self-adaptive systems (SAS), necessitating efficient verification methods.

Q2 Problem Context: 2
The context is explicitly described as the need for perpetual satisfaction of non-functional requirements (e.g., reliability, energy) in SAS operating in unpredictable environments, using models at runtime.

Q3 Research Design: 2
The research design is explicitly described, involving a two-step approach (design-time pre-computation, run-time evaluation), formal foundations (DTMCs, PCTL), algorithms, implementation (WM tool), and empirical evaluation.

Q4 Contributions: 2
Contributions are explicitly listed: a mathematical framework for run-time efficient probabilistic model checking, support for sensitivity analysis, and an extended evaluation.

Q5 Insights: 2
The paper provides explicit insights, such as the trade-offs between matrix-based and equation-based methods, the impact of the number of parameters, and the significant runtime speedup achieved.

Q6 Limitations: 2
Limitations are explicitly discussed in Sections 4, 10.5, and elsewhere, including assumptions about the number of variable parameters, the practicality for nested formulae, and numerical issues.

Total Score: 12 / 12

---

31. Title: TARL: Modeling Topology Adaptations for Networking Applications

Q1 Problem Definition: 2
Explicitly defines the problem: the lack of a well-defined, reusable model for topology adaptations in

networking applications, leading to tightly integrated, hard-to-understand, compare, and reuse implementations.

Q2 Problem Context: 2
Explicitly describes the context of two application domains (Video Streaming Overlays and Wireless Sensor Networks) and their need for decentralized topology adaptations, deriving five key characteristics.

Q3 Research Design: 2
Explicitly describes the research design: analysis of 14 existing adaptations, derivation of a general model, design of the TARL language, implementation of a runtime framework, and evaluation via case studies and expressiveness analysis.

Q4 Contributions: 2
Explicitly lists contributions: a general model for topology adaptations, the TARL rule language, a runtime environment, and an evaluation showing expressiveness.

Q5 Insights: 2
Provides explicit insights, such as the five key characteristics of topology adaptations, the expressiveness of TARL for 13/14 algorithms, and the feasibility demonstrated by porting an existing application.

Q6 Limitations: 2
Explicitly discusses limitations: TARL cannot express the LMST algorithm due to its requirement for matching paths of arbitrary length, and the interpreter-based runtime is infeasible for resource-constrained WSNs (requiring a compiler under development).

Total Score: 12 / 12

---

32. Title: Towards Digital Twin-enabled DevOps for CPS providing Architecture-Based Service Adaptation & Verification at Runtime

Q1 Problem Definition: 2
Explicitly defines the problem of uncertainty in CPS/IPSS design and operations, and the need for continuous adaptation akin to IT DevOps.

Q2 Problem Context: 2
Provides explicit context of Cyber-Physical Systems (CPS), Industrial Product-Service Systems (IPSS), and the specific challenges of Operation Technology (OT) vs. Information Technology (IT) environments.

Q3 Research Design: 1
Describes the method generally (transferring DevOps principles, using design requirements, proposing models, and evaluation), but lacks an explicit, step-by-step research methodology section.

Q4 Contributions: 2
Explicitly states contributions: a self-adaptive CPS model, a DT design concept, and a novel A/B testing deployment strategy for OT.

Q5 Insights: 2
Provides explicit insights from the evaluation, e.g., achieving sub-millisecond cycle times during A/B testing without downtime, and discusses the feasibility of the proposed models.

Q6 Limitations: 1
Discusses limitations generally (e.g., implementation complexity, increased resource needs, need for future evaluation under uncertainty) but not in a dedicated, structured limitations section.

Total Score: 10 / 12

---

33. Title: Towards Secure Architecture-based Adaptations

Q1 Problem Definition: 2
Explicitly defines the problem: self-adaptation can introduce new security threats and vulnerabilities, and little research has been devoted to analyzing the security risks of architecture-based adaptations at runtime.

Q2 Problem Context: 2
Explicitly describes the context of architecture-based self-adaptive systems, specifically using the Rainbow framework and its Stitch/Arme languages for modeling and adaptation.

Q3 Research Design: 2
Provides an explicit description of the methodology: using threat modeling/attack graphs, integrating a security dimension into Rainbow's MAPE-K loop, defining security metrics, and evaluating with two case studies.

Q4 Contributions: 2
Explicitly lists three contributions: a generic approach for vulnerability analysis, its implementation in Rainbow, and an application/evaluation on two cases.

Q5 Insights: 2
Provides explicit insights from the evaluation, showing how different weights for the security dimension lead to the selection of more secure adaptations, and discusses the trade-offs.

Q6 Limitations: 1
Discusses limitations generally (e.g., small number of experiments, need for study on more complex systems, computational complexity) within the evaluation section, but not in a dedicated "Limitations" subsection.

Total Score: 11 / 12

---

34. Title: Towards Reusability in Autonomic Computing

Q1 Problem Definition: 2
Explicitly defines the problem: reusability in autonomic/self-adaptive system development is insufficient, forcing developers to start from scratch, and existing frameworks neglect reusability at the component implementation level.

Q2 Problem Context: 2
Provides explicit context of Autonomic Computing, self-adaptive systems (SAS), the MAPE-K model, and the distinction between adaptation logic (AL) and managed resources.

Q3 Research Design: 2
Explicitly describes the methodology: proposing a reusable AL template and component template, a pub/sub communication architecture, and evaluating with two case studies (smart highway, data center) measuring Lines of Code (LOC) reuse.

Q4 Contributions: 2
Explicitly lists three contributions: a template for the AL based on MAPE, a reusable decentralized communication structure, and an evaluation showing how reusability speeds up development.

Q5 Insights: 2
Provides explicit, quantitative insights from the evaluation, showing up to 94.1% code reuse between different use cases and analyzing reusability at the component and system level.

Q6 Limitations: 1
Discusses limitations indirectly as future work (e.g., hard-coded logic loading, limited pattern integration, need for detailed communication evaluation) but not in a dedicated "Limitations" section.

Total Score: 11 / 12

---

35. Title: mRUBiS: An Exemplar for Model-Based Architectural Self-Healing and Self-Optimization

Q1 Problem Definition: 2
Explicitly defines the problem: existing exemplars do not provide architectural runtime models, forcing developers to implement them from scratch, which is a costly and complex task.

Q2 Problem Context: 2
Provides explicit context of model-based architectural self-adaptation, the MAPE-K loop, causal connection, and a survey of related exemplars and their limitations.

Q3 Research Design: 2
Explicitly describes the design of the mRUBiS exemplar, including the generic simulation framework,

the CompArch modeling language, and the instantiation for self-healing and self-optimization case studies.

Q4 Contributions: 2
Explicitly states the contribution: an extensible exemplar that provides an architectural runtime model and simulator to support off-the-shelf development and evaluation of model-based architectural self-adaptation.

Q5 Insights: 2
Provides explicit insights from using the exemplar, including generated utility and execution time charts, and discusses its use for early testing, evaluation, and in student projects.

Q6 Limitations: 1
Discusses limitations indirectly as future work (e.g., need for incremental utility computation, support for environment models, more case studies) but not in a dedicated limitations section.

Total Score: 11 / 12

Scopus Articles

---

36. Title: A Context-Aware Reflective Middleware Framework for Distributed Real-Time and Embedded Systems

Q1 Problem Definition: 2
The paper explicitly defines the problem: existing context-aware reflective middleware (CARM) frameworks have reconfiguration times in the range of seconds, which is too long for Distributed Real-Time and Embedded (DRE) systems requiring millisecond-level responsiveness.

Q2 Problem Context: 2
The context is explicitly described as DRE systems (e.g., vehicle safety systems, rapid response) operating in dynamic, resource-limited environments that require adaptive behavior.

Q3 Research Design: 2
The research design is explicitly described, including the design of the MARCHES framework (architecture, reflection model, synchronization protocol) and its evaluation through a theoretical analytical model and empirical measurements.

Q4 Contributions: 2
Contributions are explicitly stated: a new CARM framework (MARCHES) with a multiple component-chain architecture and an active-message-based synchronization protocol to drastically reduce reconfiguration time, plus a novel analytical model for comparison.

Q5 Insights: 2
Insights from the evaluation are explicit: reconfiguration time is reduced from seconds to hundreds of microseconds, with low overhead, good scalability, and small memory footprint, making it suitable for DRE systems.

Q6 Limitations: 2
Limitations are explicitly stated: the framework focuses on stateless applications, and its component model currently only supports COM and .NET components, with support for others (e.g., CORBA, Java Beans) left as future work.

Total Score: 12 / 12

---

37. Title: A Feature-Oriented Approach for Developing Reusable Product Line Assets of Service-Based Systems

Q1 Problem Definition: 2
The paper explicitly defines the core problem: the practical challenges of developing reusable assets for service-based systems, specifically the identification of reusable services, determination of context-relevant configurations, and maintenance of service validity after configuration changes.

Q2 Problem Context: 2
The context is explicitly described as Service-Oriented Architectures (SOA) and Software Product Line Engineering (SPLE) for developing flexible, context-aware systems that must adapt to changing user needs and dynamic environments (e.g., the "Virtual Office of the Future" case study).

Q3 Research Design: 2
The research design is explicitly described, involving the adaptation of a feature-oriented product line engineering approach, detailed steps for feature/service/context analysis, the proposal of the HEART architecture model, and validation through a case study and prototype implementation.

Q4 Contributions: 2
Contributions are explicitly stated: a method for identifying reusable services at the right granularity, mapping user context to service configuration, checking service validity at runtime, and a heterogeneous style-based architecture model (HEART) for systematic development.

Q5 Insights: 2
Insights from applying the method are explicit, demonstrated through the case study and prototype, showing how the approach addresses the identified challenges and enables the construction of flexible, reusable service-based systems.

Q6 Limitations: 1
Limitations are discussed in a general way within the conclusion and future work sections (e.g., the approach's current focus, the ongoing work to establish a full demonstration facility), but not in a dedicated, explicit limitations section.

Total Score: 11 / 12

---

38. Title: ASPLe: A Methodology to Develop Self-Adaptive Software Systems with Systematic Reuse

Q1 Problem Definition: 2
The problem is explicitly defined as the lack of dedicated process support to design and develop self-adaptive software systems (SASS) with systematic reuse, which hinders knowledge transfer and quality design.

Q2 Problem Context: 2
The context is explicitly described as the intersection of Self-Adaptive Software Systems (SASS) and Software Reuse (specifically Software Product Line Engineering), focusing on managing uncertainty and runtime variability.

Q3 Research Design: 2
The research design is explicitly detailed: proposing the ASPLe methodology (comprising three processes: Domain Engineering, Specialization, Integration), each with sub-processes, and evaluating it via a case study with quantitative and qualitative methods.

Q4 Contributions: 2
Contributions are explicitly stated: the ASPLe methodology providing step-by-step process support, the extended Architectural Reasoning Framework (eARF), and evaluation results showing positive effects on reuse and quality.

Q5 Insights: 2
Insights from the case study are explicitly discussed, showing statistically significant improvements in total reuse and fault density, and qualitative feedback on the methodology's strengths (e.g., separation of concerns, variability management).

Q6 Limitations: 2
Limitations are explicitly stated in the conclusion: ASPLe currently supports requirements and design activities but needs extension to implementation and testing; it requires more evaluation in real-world settings; and descriptions/examples could be improved.

Total Score: 12 / 12

---

39. Title: A Domain-Specific Language for the Control of Self-Adaptive Component-Based Architecture

Q1 Problem Definition: 2
The paper explicitly defines the problem of ensuring correct navigation through configuration spaces in self-adaptive component-based systems and the low-level, error-prone nature of programming fine-grained reconfiguration actions.

Q2 Problem Context: 2
The context of component-based software engineering, Architecture Description Languages (ADLs), and the need for formal guarantees in self-adaptive systems is explicitly described.

Q3 Research Design: 2
The research design is explicitly described, including the proposal of the Ctrl-F DSL, its formal translation to Heptagon/BZR, the integration with the FraSCAti platform, and the evaluation via two case studies.

Q4 Contributions: 2
Contributions are explicitly listed: the Ctrl-F language, its compilation to Heptagon/BZR, its integration with a middleware, and the application to case studies demonstrating predictive control.

Q5 Insights: 2
Insights from the case studies and a discussion on applicability, limitations (e.g., scalability), and comparison with other approaches are explicitly provided.

Q6 Limitations: 2
Limitations are explicitly discussed in Section 6.3, including the combinatorial complexity of Discrete Controller Synthesis (DCS) and its impact on scalability.

Total Score: 12 / 12

---

40. Title: Analyzing Self-Adaptive Systems as Software Product Lines

Q1 Problem Definition: 2
Explicit problem definition. The paper clearly states the challenge of managing runtime variability in Self-Adaptive Systems (SAS) and proposes modelling them as Dynamic Software Product Lines (DSPLs) to leverage formal analysis techniques.

Q2 Problem Context: 2
Explicit context description. The context of SAS operating under uncertainty, particularly in physical environments like robotics, is thoroughly described, including the two-layered (managed/managing) architecture and the use of MAPE-K loops.

Q3 Research Design: 2
Explicit description of how research was done. The methodology is clearly outlined: proposing a correspondence between SAS and DSPLs, demonstrating it via a small-scale evaluation (AUV case study), modelling with ProFeat, and performing specific family-based analyses.

Q4 Contributions: 2
Explicit contributions description. Contributions are listed in a dedicated subsection (end of Section 1),

including extending previous work with new background, extended evaluation, models, analyses, and evaluation of extensibility.

Q5 Insights: 2
Explicit insights description. The evaluation section (Section 6) explicitly answers three research questions regarding extensibility, analysis capabilities, and verification of adaptation logic, providing clear insights from the case study.

Q6 Limitations: 2
Explicit limitations description. A dedicated "Threats to validity" subsection (6.4) discusses limitations in scope (other model extensions, scalability), construct validity, and researcher bias.

Total Score: 12 / 12

---

41. Title: BASBA: A Framework for Building Adaptable Service-Based Applications

Q1 Problem Definition: 2
Explicit problem definition: Developing adaptive service-based applications (SBAs) is complex, costly, and often unsystematic with adaptation logic interwoven with application logic.

Q2 Problem Context: 2
Explicit context description: SBAs operate in dynamic, unpredictable environments requiring QoS adherence; current approaches lack systematic adaptation mechanisms and reuse.

Q3 Research Design: 2
Explicit description: Research design includes proposing the BASBA framework (metamodel, reusable tactics), model transformations, and empirical evaluation via two case studies and an academic study.

Q4 Contributions: 2
Explicit contributions: A model-based framework with reusable adaptation tactics, runtime models, and empirical evidence of improved development time and quality.

Q5 Insights: 2
Explicit insights: Statistical results and discussion on BASBA's effectiveness, benefits for non-professional developers, and limitations in complex business scenarios.

Q6 Limitations: 2
Explicit limitations: Section 7.5 discusses limitations of BASBA, including limited tactic library, inefficiency for highly complex/domain-specific adaptations, and varied effectiveness for professional developers.

Total Score: 12 / 12

---

42. Title: Incorporating Architecture-Based Self-Adaptation into an Adaptive Industrial Software System

Q1 Problem Definition: 2
Explicit problem definition: Traditional resilience mechanisms (embedded low-level code or human oversight) are inadequate; the effectiveness of Architecture-Based Self-Adaptation (ABSA) in industrial-scale systems with legacy commitments needs evaluation.

Q2 Problem Context: 2
Explicit context description: The context is a large-scale commercial middleware (DCAS) for monitoring and managing networks of devices in renewable energy plants, which already has embedded adaptive mechanisms and human-operated adaptations.

Q3 Research Design: 2
Explicit description: Research design includes integrating the Rainbow ABSA framework with DCAS, re-implementing existing adaptations, extending with new ones, and evaluating implementation effort and operational performance through experiments.

Q4 Contributions: 2
Explicit contributions: An experience report demonstrating that ABSA can replicate and improve upon existing adaptations, with quantitative data on development effort, performance benefits, and lessons learned about reuse and evolution.

Q5 Insights: 2
Explicit insights: Section 6 provides detailed lessons learned on development paradigms, information use, scope of ABSA, monitoring infrastructure, and evolution effort, supported by experimental results.

Q6 Limitations: 2
Explicit limitations: Threats to validity are discussed in Section 7; limitations of the approach (e.g., fixed monitoring infrastructure, need to choose what to manage with ABSA) are explicitly stated in the lessons learned (Section 6).

Total Score: 12 / 12

---

43. Title: MOO: An Architectural Framework for Runtime Optimization of Multiple System Objectives in Embedded Control Software

Q1 Problem Definition: 2
Explicitly defines the problem: lack of systematic methods for designing and implementing Multi-Objective Optimization (MOO) leads to ad hoc, tightly coupled, non-reusable solutions.

Q2 Problem Context: 2
Explicitly describes the context of embedded control software with a detailed industrial case study (Warm Process in printing systems).

Q3 Research Design: 2
Explicitly describes the research design, including the MOO architectural style, toolchain components, and experimental setup for validation.

Q4 Contributions: 2
Explicitly states the contribution: a complete architectural framework (style, editor, tools, code generator) for MOO.

Q5 Insights: 2
Provides explicit insights from the experiment, showing improved performance, modularity, and reusability compared to state-of-practice solutions.

Q6 Limitations: 2
Explicitly states current limitations of the framework (static architectural view, lack of probabilistic constraints) in the "Conclusion and Future Work" section.

Total Score: 12 / 12

---

44. Title: Virtualized Web Server Cluster Self-Configuration to Optimize Resource and Power Use

Q1 Problem Definition: 2
Explicitly defines the problem of energy waste in data centers due to idle resources and the need for resource optimization and quality of service in virtualized web server clusters.

Q2 Problem Context: 2
Explicitly describes the context of virtualization technology, server consolidation, and existing limitations of vendor-dependent APIs for managing virtualized environments.

Q3 Research Design: 2
Explicitly describes the development of a prototype architecture, the creation of two different managers (Single Step and Composed Step), and the performance evaluation methodology using specific workloads and metrics.

Q4 Contributions: 2
Explicitly states the contribution of a reusable architecture, a high-level API (HVMM), and two resource managers for self-configuration aimed at QoS and energy saving.

Q5 Insights: 2
Provides explicit insights from the performance evaluation, comparing the two managers' effectiveness in reducing power consumption and maintaining response time.

Q6 Limitations: 1
Discusses limitations generally in the "Conclusion and Future Work" (e.g., focusing only on CPU, fixed number of VMs) but not in a dedicated, explicit limitations section.

Total Score: 11 / 12

---

45. Title: Self-Adaptive Architecture Evolution with Model Checking: A Software Cybernetics Approach

Q1 Problem Definition: 2
Explicitly defines the problem: controlling the emergent, uncertain behavior of self-adaptive architectures and detecting/fixing faults in adaptation rules.

Q2 Problem Context: 2
Explicitly describes the context of cloud computing and self-adaptive systems, with a detailed running example (smartphone application).

Q3 Research Design: 2
Explicitly describes the research design, including the Breeze/ADL language, the cybernetics-based control framework, the learning algorithm, and the experimental case study.

Q4 Contributions: 2
Explicitly lists contributions in the introduction, including the novel framework, the use of Breeze/ADL, the learning algorithm, and tool extensions.

Q5 Insights: 2
Provides explicit insights from the evaluation, including the effectiveness of the framework in detecting error states and the reduction of error states after applying feedback.

Q6 Limitations: 2
Explicitly discusses threats to validity in a dedicated section (Section 7.6), covering small scale, limited subjects, and limited constraint types.

Total Score: 12 / 12