SKILLFORGE

A Modular, Skills-Based Architecture for Democratized AI Agent Construction

A Technical and Commercial Research Paper

Prepared by

Nuqta Technology LLC

Point Technologies LLC • CR: 1628390

Muscat, Sultanate of Oman

April 2026

Version 1.3

ABSTRACT

The proliferation of large language models (LLMs) has catalyzed an unprecedented transformation in human-computer interaction, yet a fundamental asymmetry persists: while the underlying models grow increasingly capable, the mechanisms through which non-technical users harness this capability remain primitive, fragmented, and inaccessible. This paper introduces **SkillForge**, a novel platform architecture that reimagines AI agent construction through a modular, skills-based paradigm. Rather than requiring users to engage in prompt engineering, API orchestration, or code-level customization, SkillForge enables the assembly of sophisticated AI agents through the composition of discrete, validated, and interoperable skill modules.

The proposed architecture decomposes agent capability into atomic skill units—each encapsulating domain knowledge, behavioral patterns, tool integrations, and output modalities—that can be combined, configured, and deployed by users of varying technical proficiency. This paper provides a rigorous technical specification of the SkillForge architecture, including the **Skill Definition Language (SDL)**, the **Agent Composition Engine (ACE)**, the **Runtime Execution Framework (REF)**, and the **Skill Marketplace Protocol (SMP)**. We present a comprehensive analysis of the underlying AI model landscape, with particular emphasis on Claude Sonnet 4.6 and Claude Opus 4.6 as foundational inference engines, including detailed cost modeling, latency benchmarking, and quality-cost trade-off analysis.

Furthermore, this paper situates SkillForge within the broader commercial and geopolitical landscape, examining the global AI agent market—projected to reach USD 10.91 billion in 2026 and USD 182.97 billion by 2033—and the specific strategic opportunity within the Sultanate of

Oman, whose National Programme for AI and Advanced Digital Technologies (2024–2026) and Vision 2040 framework create a uniquely receptive environment for localized AI innovation. We present a competitive analysis distinguishing SkillForge from existing solutions including ChatGPT, Claude, Microsoft Copilot Studio, and Salesforce Agentforce, and articulate the theoretical foundations drawn from software engineering, cognitive science, and economics that underpin the skills-based approach.

The paper concludes with a phased implementation roadmap, business model analysis, risk assessment, and directions for future research in skill composability, multi-agent orchestration, and Arabic-first AI agent frameworks.

Keywords: *AI Agents; Skills-Based Architecture; Large Language Models; Agent Composition; Platform Economics; Oman Vision 2040; Modular AI; Claude API; Democratized AI; Agentic AI; Skill Marketplace; Arabic NLP*

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

Table 1: Claude API Model Tiers and Pricing (April 2026)
Table 2: Cost Optimization Strategies and Projected Savings
Table 3: Monthly Cost Model for 1,000-Agent Deployment
Table 4: Comparative LLM Analysis for Agent Platform Use
Table 5: Global AI Agent Market Projections by Research Firm
Table 6: Oman Digital Economy Key Metrics
Table 7: Competitive Differentiation Matrix
Table 8: SDL Schema Structure Summary
Table 9: Execution Pipeline Stages
Table 10: Risk Assessment Matrix
Table 11: Revenue Model Projections (Year 1–3)
Figure 1: SkillForge Layered Architecture Overview (described)
Figure 2: Agent Composition Engine Data Flow (described)
Figure 3: Execution Pipeline Sequence Diagram (described)
Figure 4: Cost-Quality Pareto Frontier (described)
Figure 5: Skill Marketplace Ecosystem Model (described)

1. INTRODUCTION

1.1 The Paradox of AI Accessibility

The year 2026 marks a peculiar inflection point in the history of artificial intelligence. Never before have AI models been so powerful, so versatile, and so widely available—and yet, never before has the gap between what AI can theoretically accomplish and what most people can actually make it accomplish been so vast. The global AI agents market, valued at approximately USD 7.63 billion in 2025, is projected to reach USD 10.91 billion in 2026, representing a 43% year-over-year increase—the steepest growth curve in enterprise software since the cloud computing revolution.[1]

These figures represent not merely an industry's growth trajectory but a fundamental reorganization of how knowledge work is performed, how businesses operate, and how individuals interact with computational systems. Projections for the end of the decade and beyond are even more striking: Grand View Research estimates the market will reach USD 182.97 billion by 2033, growing at a compound annual growth rate (CAGR) of 49.6%.[2]

Yet beneath these staggering numbers lies an uncomfortable truth: the vast majority of this value is being captured by a small number of technology companies and their most sophisticated enterprise clients. The individual user, the small business owner, the educator, the healthcare provider, the government administrator—all of whom stand to benefit enormously from AI agent technology—remain largely locked out of meaningful agent customization. They can use ChatGPT or Claude as general-purpose conversational interfaces, but they cannot construct purpose-built agents that reflect their unique domain expertise, workflow requirements, and operational constraints.

[1]Grand View Research, "AI Agents Market Size and Share: Industry Report, 2033," 2026.
[2]Ibid. CAGR of 49.6% from 2026 to 2033.

This paper addresses this fundamental asymmetry by proposing **SkillForge**: a platform architecture that transforms AI agent construction from a specialized engineering discipline into an accessible, composable, and democratized activity. The core insight is that agent capability can be decomposed into discrete **skill modules**—standardized, validated, and interoperable units of domain expertise—that can be assembled by non-technical users into arbitrarily sophisticated agents.

1.2 Motivation and Research Questions

The motivation for this research emerges from three convergent observations:

First, the increasing maturity of foundation models—particularly the Claude 4.6 family from Anthropic and competing models from OpenAI, Google DeepMind, and Meta—has made sophisticated natural language understanding and generation a commodity rather than a differentiator. The transformer architecture, introduced by Vaswani et al. in 2017, has been scaled to models with hundreds of billions of parameters that demonstrate emergent capabilities in reasoning, planning, and multi-step task execution.[3]

Second, the emergence of "agentic AI" as a distinct paradigm, where AI systems are expected not merely to respond to queries but to reason, plan, and execute multi-step tasks autonomously, creates an urgent need for structured agent composition frameworks. By 2026, approximately 40% of enterprise applications are expected to include task-specific AI agents, and 51% of large enterprises report having AI agents in production.[4]

Third, the specific strategic context of the Sultanate of Oman, with its ambitious Vision 2040 digital transformation agenda, its National Programme for AI and Advanced Digital

[3]Vaswani, A. et al. (2017). "Attention Is All You Need." NeurIPS 30.
[4]Gartner, "Top Strategic Technology Trends for 2026." Projects 40% enterprise apps will include AI agents.

Technologies (2024–2026), and its newly unveiled 2026–2030 Digital Economy Roadmap, presents a compelling case study for how a skills-based agent platform can serve both local needs and global markets.[5]

This paper addresses the following research questions:

RQ1: How can AI agent construction be decomposed into modular, composable skill units that enable non-technical users to build sophisticated agents without programming expertise?

RQ2: What architectural patterns and technical frameworks are required to support skill composition, runtime execution, and quality assurance in a multi-tenant agent platform?

RQ3: What is the optimal model selection and cost optimization strategy for a skills-based agent platform utilizing contemporary large language models?

RQ4: How does the global and regional (Omani/GCC) market landscape create commercial opportunities for such a platform?

RQ5: What theoretical foundations from computer science, cognitive science, and economics justify the skills-based approach over alternatives?

1.3 Contributions

This paper makes the following contributions:

(i) We introduce the **Skill Definition Language (SDL)**, a formal specification for encoding domain expertise, behavioral patterns, tool integrations, and output modalities into discrete, versioned, and composable skill modules.

[5]MTCIT, Sultanate of Oman, "Oman AI & Digital Future Program (2024–2026)," 2024.

(ii) We present the **Agent Composition Engine (ACE)**, a runtime framework that orchestrates skill execution, manages inter-skill dependencies, handles context propagation, and enforces quality constraints across composed agents.

(iii) We provide the first comprehensive **cost modeling analysis** for a multi-model agent platform built on the Claude API, including comparisons across Claude Opus 4.6, Sonnet 4.6, and Haiku 4.5, with prompt caching, batch processing, and extended thinking optimizations.

(iv) We present a **market analysis** situating the platform within Oman's digital transformation ecosystem, including alignment with the National AI Programme and the 2026–2030 Roadmap.

(v) We articulate the **theoretical foundations** for the skills-based approach, drawing on component-based software engineering, distributed cognition theory, and platform economics.

(vi) We provide a **competitive analysis** distinguishing SkillForge from ChatGPT, Claude, Copilot Studio, Agentforce, and open-source alternatives.

(vii) We propose a **business model** with unit economics analysis demonstrating commercial viability at per-interaction costs of approximately USD 0.023.

1.4 Paper Organization

The remainder of this paper is organized as follows. **Section 2** presents the theoretical foundations and literature review. **Section 3** defines the problem space and articulates design requirements. **Section 4** details the SkillForge architecture. **Section 5** addresses model selection and cost analysis. **Section 6** provides the market and competitive analysis, with particular emphasis on the Omani opportunity. **Section 7** discusses implementation methodology. **Section 8** presents the business model and unit economics. **Section 9** addresses risks, limitations, and

ethical considerations. **Section 10** outlines future research directions. **Section 11** concludes the paper.

2. THEORETICAL FOUNDATIONS AND LITERATURE REVIEW

2.1 Component-Based Software Engineering

The intellectual genealogy of SkillForge traces directly to the principles of component-based software engineering (CBSE), a paradigm that emerged in the late 1990s and fundamentally transformed how complex software systems are designed, built, and maintained. Szyperski defined a software component as "a unit of composition with contractually specified interfaces and explicit context dependencies only," a definition that maps precisely onto our concept of a skill module.[6]

The key insight of CBSE is that complex systems can be assembled from pre-built, tested, and reusable components, dramatically reducing development time, improving reliability, and enabling a division of labor between component developers and system assemblers. In the context of AI agent construction, this principle translates to a separation between **skill developers**—who encode domain expertise, behavioral patterns, and tool integrations into standardized skill modules—and **agent builders**—who compose these modules into purpose-built agents without needing to understand the internal implementation details.

The success of component-based approaches in modern software is well-documented and provides strong precedential support for the SkillForge model. The npm ecosystem for JavaScript contains over 2.5 million packages, each representing a reusable component that can be composed into larger systems. Docker containers and Kubernetes orchestration have demonstrated how complex distributed systems can be assembled from modular, standardized units. React's component model has shown how even user interface construction—traditionally a highly bespoke activity—can be decomposed into reusable, composable units with well-defined

[6]Szyperski, C. (2002). Component Software: Beyond Object-Oriented Programming, 2nd ed. Addison-Wesley.

property interfaces and lifecycle methods. SkillForge applies this same philosophy to AI agent construction, treating skills as the fundamental unit of composition.

2.2 Distributed Cognition and Extended Mind Theory

The theoretical justification for a skills-based agent architecture extends beyond software engineering into cognitive science. The theory of distributed cognition, as articulated by Hutchins, posits that cognitive processes are not confined to individual minds but are distributed across individuals, artifacts, and environmental structures.[7]

In this framework, an AI agent composed of multiple skill modules can be understood as a distributed cognitive system, where each skill module contributes specialized knowledge and processing capabilities that, when orchestrated together, produce emergent cognitive capabilities exceeding those of any individual component. This is analogous to how a ship's navigation team, in Hutchins's seminal study, distributes cognitive labor across multiple individuals and instruments, achieving navigational precision that no single team member could accomplish alone.

Clark and Chalmers extended this perspective with their Extended Mind hypothesis, arguing that cognitive processes can legitimately extend beyond the biological boundary of the brain to incorporate external tools and resources.[8]

A skills-based AI agent, in this view, represents a form of **extended cognition** where the human user's cognitive capabilities are augmented by a carefully curated and composed set of AI skills. The skills-based approach—rather than a monolithic agent—preserves the user's agency in selecting, combining, and configuring the cognitive extensions they employ, maintaining what

[7]Hutchins, E. (1995). Cognition in the Wild. MIT Press.
[8]Clark, A. and Chalmers, D.J. (1998). "The Extended Mind." Analysis, 58(1), 7–19.

we term **"cognitive sovereignty."** The user remains the architect of their own cognitive augmentation, choosing which skills to deploy and how they interact, rather than being presented with a fixed, predetermined set of capabilities.

2.3 Platform Economics and Two-Sided Markets

The commercial viability of SkillForge is grounded in the economics of platforms and two-sided markets, as formalized by Rochet and Tirole and further developed by Parker, Van Alstyne, and Choudary.[9]

A skills marketplace creates a classic **two-sided market** connecting skill developers (supply side) with agent builders (demand side), with the platform extracting value by facilitating transactions, maintaining quality standards, and providing the infrastructure for skill execution. The platform economics framework predicts several dynamics critical to SkillForge's commercial strategy:[10]

Network Effects: Both direct effects (more users attract more users through social proof and knowledge sharing) and indirect effects (more skill developers attract more agent builders by expanding available capabilities, and more agent builders attract more skill developers by expanding the addressable market) create powerful competitive moats once critical mass is achieved. The strength of these network effects scales superlinearly with platform size, creating a winner-take-most dynamic that favors early entrants who achieve critical mass.

Zero Marginal Cost Distribution: Once a skill is developed and published, the marginal cost of distributing it to additional users approaches zero. This enables scale economics fundamentally different from traditional software services, where each new customer typically

[9]Rochet, J.C. and Tirole, J. (2003). "Platform Competition in Two-Sided Markets." JEEA, 1(4), 990–1029.
[10]Parker, G., Van Alstyne, M., and Choudary, S.P. (2016). Platform Revolution. W.W. Norton.

incurs incremental infrastructure and support costs. For skills, the only variable cost is the AI model inference cost, which is borne by the end user and optimized through the platform's caching and routing infrastructure.

Value Creation Through Composition: Unlike traditional marketplaces where products are consumed individually, the skills marketplace creates additional value through composition. A medical skill and a document formatting skill, purchased separately, create value greater than the sum of their parts when composed into an agent that can produce formatted medical reports. This compositional value creation is a unique economic property of the skills-based approach.

2.4 The Agentic AI Paradigm

The emergence of agentic AI as a distinct paradigm represents a fundamental shift from AI as a passive tool to AI as an autonomous or semi-autonomous collaborator. This shift is driven by several technological advances: chain-of-thought prompting, which enables multi-step reasoning; the ReAct framework, which synergizes reasoning with action; and tool-use capabilities, demonstrated by systems like Toolformer, which enable language models to invoke external tools and APIs.[11]

The agentic paradigm introduces several architectural requirements that directly inform SkillForge's design:[12]

Multi-Step Reasoning: Agents must decompose complex tasks into subtasks, each of which may require different skills or tools. A user who asks an agent to "research recent cybersecurity threats in Oman and prepare a briefing document" is requesting a multi-step

[11]Wei, J. et al. (2022). "Chain-of-Thought Prompting Elicits Reasoning in LLMs." NeurIPS 35.
[12]Yao, S. et al. (2023). "ReAct: Synergizing Reasoning and Acting in Language Models." ICLR 2023.

workflow that spans web research, information synthesis, threat analysis, and document generation—each requiring distinct skills.[13]

State Maintenance: Agents must maintain context across interactions, remembering the user's preferences, prior conversations, and ongoing tasks. This state management is distributed across skills in SkillForge, with each skill maintaining its own context slice and a shared context bus coordinating cross-skill state.

Tool Use: Agents must be capable of invoking external services, APIs, and data sources as part of their task execution. In SkillForge, tool use is encapsulated within skills, with each skill declaring its tool integrations in its SDL definition.

Safety and Governance: Autonomous agent behavior must remain aligned with user intent and organizational policy. SkillForge addresses this through skill-level safety contracts and an agent-level governance layer that enforces configurable constraints on agent behavior.

2.5 Prior Art in Agent Construction Platforms

Several existing platforms and frameworks have addressed aspects of the agent construction problem. We review the most significant prior art and identify the gaps that SkillForge addresses.

OpenAI GPTs (launched 2023, expanded 2024–2025) introduced user-configurable agents through natural language instructions, knowledge file uploads, and action integrations. While pioneering in accessibility, GPTs lack modular composability; each GPT is configured as a monolithic entity rather than composed from reusable skill components. There is no mechanism for a GPT's configuration elements to be shared, versioned, or composed with other GPTs.

[13]Schick, T. et al. (2023). "Toolformer: Language Models Can Teach Themselves to Use Tools." NeurIPS 36.

Microsoft Copilot Studio (November 2024) provides a low-code platform with drag-and-drop features for creating agents. While demonstrating market demand for accessible agent construction, its architecture is tightly coupled to the Microsoft ecosystem—requiring Microsoft 365 licensing and integrating primarily with Microsoft services—limiting portability and cross-platform utility.

Salesforce Agentforce leads enterprise performance benchmarks with an average composite score of 9.5 across technical, operational, financial, and governance metrics. However, it is purpose-built for CRM use cases and does not provide a general-purpose skill composition framework accessible to users outside the Salesforce ecosystem.[14]

LangChain (115,000+ GitHub stars) has become the de facto standard for LLM application development, providing abstractions for chains, agents, and tool use. However, LangChain is a developer framework, not an end-user platform; it requires substantial Python expertise and does not provide marketplace or composition features for non-technical users.

AutoGPT (178,000+ GitHub stars) and **CrewAI** represent community-driven approaches to autonomous agent construction. These frameworks demonstrate strong demand but require significant technical expertise and do not provide the structured quality assurance, marketplace economics, or production-grade reliability that enterprise and SME adoption requires.

The gap in the existing landscape is clear: **no current platform provides a comprehensive skills-based architecture that combines modular composability, marketplace economics, non-technical accessibility, multi-model optimization, and production-grade reliability.** SkillForge is designed to fill this gap.

[14]DemandSage, "AI Agents Market Size, Share & Trends (2026–2034 Data)," January 2026.

3. PROBLEM DEFINITION AND DESIGN REQUIREMENTS

3.1 Formal Problem Statement

We formally define the **Agent Construction Problem (ACP)** as follows:

Given a user U with a set of domain requirements D = {d_1, d_2, ..., d_n}, a set of desired capabilities C = {c_1, c_2, ..., c_m}, and a set of operational constraints O = {o_1, o_2, ..., o_p} (including budget, latency, privacy, and governance requirements), construct an AI agent A that satisfies all elements of D, C, and O while minimizing the technical expertise required of U and maximizing the quality, reliability, and cost-efficiency of A.

The ACP is NP-hard in the general case, as the space of possible agent configurations grows combinatorially with the number of available skills, configuration parameters, and model choices. For a marketplace with n skills, each with k configuration parameters, the search space is $O(2^n \times k^n)$, which becomes intractable for even modest values of n and k. However, practical heuristics—including skill compatibility constraints, user preference modeling, template-based starting points, and AI-assisted composition—reduce the effective search space to manageable dimensions.

3.2 Taxonomy of Current Limitations

Existing approaches to the ACP suffer from five fundamental limitations that SkillForge addresses:

Limitation 1: The Monolithic Agent Problem. Current platforms provide monolithic agents with fixed or minimally configurable capability sets. A user who needs an agent combining medical knowledge with Arabic language proficiency, document formatting, and regulatory compliance awareness must rely on the base model's general training, with no

mechanism to enhance or guarantee specific capabilities. The result is agents that are generalists when specialists are needed.

Limitation 2: The Expertise Barrier. Platforms that allow meaningful customization (LangChain, AutoGPT, custom API integrations) require substantial programming expertise. Approximately 80% of enterprise IT teams already use low-code tools for general software development, indicating strong demand for accessible construction methods, but current AI agent platforms have not adequately served this population.

Limitation 3: Integration Fragmentation. Users who need agents with multiple capabilities must typically integrate multiple tools, APIs, and services manually, creating fragile systems with inconsistent interfaces, duplicated context management, and unpredictable behavior. Each integration point is a potential failure mode, and the system's reliability is bounded by its least reliable component.

Limitation 4: The Quality Assurance Gap. Custom-built agents lack standardized quality assurance mechanisms. There is no equivalent of npm's dependency resolution, Docker's container validation, or app store review processes for AI agent capabilities. Users have no reliable way to evaluate whether a custom agent configuration will perform as expected before deploying it.

Limitation 5: Ecosystem Lock-In. Current platforms lock users into specific vendor ecosystems—Microsoft for Copilot Studio, Salesforce for Agentforce, OpenAI for GPTs. Skills and configurations developed on one platform cannot be ported to another, creating switching costs that inhibit competition and innovation.

3.3 Design Requirements Specification

Based on the problem analysis, we derive eight design requirements for the SkillForge platform:

ID	Requirement	Description	Priority
DR1	Modularity	Agent capabilities decomposable into discrete, independently developed, tested, and maintained skill modules	Critical
DR2	Composability	Skill modules combinable through well-defined interfaces with automatic dependency resolution and conflict detection	Critical
DR3	Accessibility	Agent construction achievable by users with no programming expertise through intuitive visual and NL interfaces	Critical
DR4	Scalability	Support for thousands of concurrent agents, millions of daily skill executions, and tens of thousands of marketplace skills	High
DR5	Quality Assurance	Version-managed skills tested against standardized benchmarks and subject to community review	High
DR6	Cost Efficiency	Optimized model selection, prompt caching, and resource allocation minimizing per-interaction costs	High
DR7	Localizability	Arabic-first interfaces and culturally contextualized skills for Omani and GCC markets	Medium
DR8	Governance	Auditable, explainable agent behavior with configurable safety constraints	High

Table 8. Design Requirements Specification

4. THE SKILLFORGE ARCHITECTURE

4.1 Architectural Overview

SkillForge employs a four-layer architecture, each designed for independent scaling, versioning, and evolution:

Layer 1 — Skill Definition Layer: Contains the SDL specification, skill validation engine, and skill packaging system. This layer is analogous to the source code and build system in traditional software development.

Layer 2 — Agent Composition Layer: Contains the ACE, including the dependency resolver, context bus, model router, and composition validator. This layer is analogous to the linker and runtime loader in compiled software.

Layer 3 — Execution Layer: Contains the REF, including the serverless execution engine, caching infrastructure, tool sandbox, and telemetry system. This layer is analogous to the operating system and runtime environment.

Layer 4 — Marketplace Layer: Contains the skill registry, search engine, licensing system, review platform, and analytics dashboard. This layer is analogous to the package manager and app store.

[**Figure 1** describes the layered architecture. The four layers are stacked vertically with the Skill Definition Layer at the bottom and the Marketplace Layer at the top. Bidirectional arrows connect adjacent layers, representing the flow of skill definitions upward through composition and execution to marketplace distribution, and the flow of user requests downward through discovery, composition, and execution. External interfaces (Web UI, API, WhatsApp) connect to

the Execution Layer. AI Model APIs (Claude, GPT, Gemini) connect laterally to the Execution Layer.]

4.2 The Skill Definition Language (SDL)

At the foundation of SkillForge lies the Skill Definition Language, a declarative specification format that encodes all aspects of a skill module's identity, capabilities, requirements, and behavior. SDL is designed to be both human-readable and machine-parseable, using a JSON-Schema-based format with custom extensions for AI-specific constructs.

4.2.1 SDL Schema Components

A complete skill definition in SDL comprises six mandatory sections and two optional sections:

Metadata Block (mandatory): Contains the skill's UUID v4 identifier, semantic version number (following semver 2.0), author identity, SPDX license identifier, human-readable descriptions (short: 160 characters maximum; long: 2,000 characters maximum), marketplace categorization tags, and a compatibility matrix specifying supported models, natural languages, and minimum runtime version. The metadata block also includes a cryptographic signature verifying the skill's authenticity and integrity.

Capability Declaration (mandatory): Formally specifies what the skill enables an agent to do. This includes typed input schemas (with supported types: string, number, boolean, file, array, and structured object), typed output schemas (with format specifications for text, structured data, files, and multimedia), and behavioral contracts (guarantees about response format, maximum latency, accuracy floor, and safety classification). The type system enables compile-time compatibility checking during agent composition, preventing type mismatches before deployment.

Knowledge Injection Protocol (mandatory): Specifies how domain knowledge is introduced into the model context. Four injection methods are supported: (a) **Static System Prompts**—fixed text prepended to every model invocation; (b) **RAG Configuration**—specifying vector store URI, embedding model, similarity threshold, and top-k retrieval parameters; (c) **Few-Shot Example Sets**—input-output pairs that demonstrate desired behavior; and (d) **Fine-Tuned Adapter References**—pointers to LoRA or QLoRA adapters that modify model behavior for specific domains. The protocol supports dynamic knowledge loading, where the skill fetches relevant context at runtime based on the specific query.

Tool Integration Manifest (mandatory): Declares external tools, APIs, and services that the skill may invoke during execution. Each tool integration includes: endpoint URI, authentication type (API key, OAuth 2.0, mTLS), rate limiting parameters (requests per period), timeout configuration, error handling strategy (retry with exponential backoff, fallback to alternative tool, graceful degradation), and response transformation schema. Tool integrations are sandboxed using a capability-based security model that prevents unauthorized network access, file system manipulation, or cross-skill data leakage.

Quality Metrics (mandatory): Defines the benchmarks and evaluation criteria against which the skill's performance is measured. Each quality metric specifies a benchmark suite (referenced by URI), a minimum passing score, and a weight indicating the metric's importance in composite quality calculations. Standard metric categories include: accuracy (correctness of factual claims), relevance (alignment with user intent), coherence (logical consistency of output), safety (absence of harmful content), and format compliance (adherence to declared output schema).

Configuration Schema (mandatory): Declares user-configurable parameters that allow agent builders to customize the skill's behavior without modifying its internal implementation. Examples include verbosity level, formality register, citation style, output language, and domain-specific parameters. Each configuration parameter has a type, default value, validation constraints, and human-readable description.

Dependency Declaration (optional): Lists other skills that this skill requires as prerequisites. Dependencies are specified with version ranges (using semver constraints), and the ACE's dependency resolver ensures that all prerequisites are satisfied before agent deployment.

Extension Points (optional): Declares hooks where other skills can inject functionality. This enables skill layering, where a base skill provides core functionality and extension skills add specialized behavior without modifying the base.

4.2.2 SDL Design Rationale: Why Skills, Not Prompts?

A critical question that this paper must address is: **why are skills necessary at all?** Why not simply use prompt templates, instruction sets, or fine-tuned models as the unit of agent composition?

The answer lies in the distinction between **specification** and **implementation**. A prompt is an implementation detail—a specific sequence of tokens designed to elicit particular behavior from a particular model. A skill is a specification—a formal declaration of capability, requirements, quality guarantees, and interfaces that is independent of any particular model or prompt. This distinction has profound implications:

Model Portability: A skill can be implemented using different prompts for different models. A "Medical Research" skill might use one prompt strategy for Claude Opus 4.6

(leveraging extended thinking) and a different strategy for Claude Sonnet 4.6 (optimized for cost). The skill's interface remains constant; only the implementation varies.

Quality Guarantees: A prompt provides no guarantees about output quality. A skill declares quality metrics and is tested against them. Agent builders can make informed decisions based on measurable quality data rather than hoping that a prompt will perform as intended.

Composability: Prompts do not compose naturally. Concatenating two prompts often produces worse results than either prompt alone, due to instruction conflicts and context dilution. Skills compose through well-defined interfaces, with the ACE managing context propagation and conflict resolution.

Versioning and Evolution: Prompts are typically unversioned strings that change without notice. Skills follow semantic versioning, enabling agent builders to lock dependencies to specific versions and receive notifications when updates are available.

4.3 The Agent Composition Engine (ACE)

The Agent Composition Engine is the core orchestration component responsible for assembling skills into functioning agents, managing runtime execution, and ensuring quality constraints are met. ACE addresses several technically challenging problems that arise when multiple independently developed skills must cooperate within a single agent.

4.3.1 Dependency Resolution

When a user composes an agent from multiple skills, ACE must resolve both explicit and implicit dependencies. **Explicit dependencies** are declared in a skill's SDL definition (e.g., a Document Formatting skill depends on a Text Generation skill). **Implicit dependencies** arise when two skills share requirements for the same context information or tool access.

ACE employs a directed acyclic graph (DAG) representation of skill dependencies and uses **topological sorting** to determine execution order. Circular dependencies are detected at composition time using Kahn's algorithm and reported as errors with diagnostic information identifying the dependency cycle. When multiple valid execution orders exist, ACE selects the order that minimizes total latency by maximizing parallelism among independent skills.

Dependency resolution also handles **version conflicts**, where two skills depend on different versions of the same prerequisite skill. ACE applies the **highest compatible version** strategy (analogous to npm's semver resolution), selecting the newest version that satisfies all declared version constraints. If no compatible version exists, the conflict is reported at composition time.

4.3.2 Context Management and the Context Bus

Perhaps the most technically challenging aspect of multi-skill agent execution is **context management**—the mechanism by which information flows between skills and between interactions. ACE implements a **context bus** architecture with the following properties:

Typed Context Slots: Each piece of context data occupies a typed slot in the context bus. Slots have names, types, producers (the skill that writes to the slot), consumers (skills that read from the slot), and retention policies (ephemeral, session-scoped, or persistent). The type system ensures that a skill reading a context slot receives data in the expected format.

Priority-Based Conflict Resolution: When multiple skills write to the same context slot, ACE uses a priority-based resolution strategy. Priority is determined by: (a) explicit priority declarations in SDL, (b) skill specificity (more specialized skills take precedence over more general ones), and (c) temporal recency (more recent writes take precedence). Users can override automatic resolution by configuring explicit priority orderings.

Context Window Optimization: The total context available to any single model invocation is bounded by the model's context window (1M tokens for Claude Opus 4.6 and Sonnet 4.6, 200K for Haiku 4.5). When the aggregated context exceeds this limit, ACE employs **progressive context pruning**: first removing ephemeral context, then summarizing verbose context, then truncating low-priority context, and finally raising an error if essential context still exceeds the window. This pruning is informed by each skill's declared context requirements in its SDL.

4.3.3 Intelligent Model Routing

Different skills have different requirements for model capability, latency, and cost. The **Model Routing Layer** in ACE maps each skill's requirements to the optimal model selection using a multi-objective optimization approach.

The routing algorithm maintains a **capability profile** for each available model, encoding its strengths and weaknesses across dimensions including: reasoning depth, code generation quality, creative writing ability, factual accuracy, Arabic language proficiency, tool-use reliability, and extended thinking capability. Each skill's SDL declares its minimum capability requirements across these dimensions.

Given a skill's capability requirements and the user's cost/latency constraints, the router computes the **Pareto frontier** of feasible model assignments—the set of assignments where no model can be substituted without degrading at least one objective. The user's trade-off preferences (expressed as a cost-quality slider in the UI) select a specific point on this frontier.

[**Figure 4** describes the Cost-Quality Pareto Frontier. The x-axis represents cost per interaction (USD), ranging from $0.001 to $0.10. The y-axis represents quality score (0–100). Three model regions are shown: Haiku 4.5 (low cost, moderate quality), Sonnet 4.6 (moderate

cost, high quality), and Opus 4.6 (higher cost, highest quality). The Pareto frontier connects the optimal points across these regions, showing that quality improvements require increasing cost investments, with diminishing returns at the high end.]

4.4 The Runtime Execution Framework (REF)

4.4.1 Execution Pipeline

When a user interacts with a SkillForge agent, the following seven-stage execution pipeline is triggered:

Stage	Component	Model Used	Description
1	Intent Classification	Haiku 4.5	Classifies user input to determine relevant skills; sub-50ms latency target
2	Skill Selection	ACE (algorithmic)	Selects optimal skill subset and computes execution DAG via topological sort
3	Context Assembly	Context Bus	Assembles conversation history, user prefs, skill knowledge, and tool credentials
4	Model Invocation	Per-skill routing	Each skill executes against its designated model with assembled context
5	Response Synthesis	Sonnet 4.6	Synthesizes individual skill outputs into coherent unified response
6	Quality Validation	Automated + model	Validates response against active skills' quality metrics; failures trigger retry
7	Delivery + Telemetry	Infrastructure	Delivers response via channel; records latency, cost, quality for analytics

Table 9. Execution Pipeline Stages

4.4.2 Caching Architecture

REF implements a four-tier caching architecture that dramatically reduces per-interaction costs:

Tier 1 — Prompt Caching: Leverages the Claude API's native prompt caching, storing previously processed system prompts and skill definitions. At standard 5-minute TTL, cache reads cost 10% of standard input price—a 90% reduction. For high-volume agents, 1-hour extended TTL (2x write cost) pays for itself after just two cache reads.

Tier 2 — Response Caching: For deterministic queries (those with identical inputs and context), previously computed responses are served directly from cache without model invocation. This is particularly effective for FAQ-type interactions and data lookup operations.

Tier 3 — Skill Output Caching: Expensive skill computations (e.g., RAG retrieval results, web research summaries) are cached at the skill level with configurable TTLs. Subsequent requests that require the same skill computation reuse cached outputs.

Tier 4 — Vector Store Caching: For RAG-based skills, embedding computations and vector similarity search results are cached to avoid redundant embedding and retrieval operations.

The combined effect of these caching tiers typically reduces per-interaction costs by **60–80%** compared to naive implementation, and reduces latency by **40–60%** for cached interactions.

4.5 The Skill Marketplace Protocol (SMP)

The Skill Marketplace is the commercial and community layer of SkillForge, enabling the creation of a two-sided market connecting skill developers with agent builders.

4.5.1 Publication and Discovery

Skills are published to the marketplace through a structured submission pipeline: (a) automated SDL validation, (b) automated quality benchmark execution, (c) security scanning for malicious tool integrations, (d) performance benchmarking on standardized hardware, and (e) community review for high-visibility skills. Published skills are indexed for both keyword and semantic search, enabling users to find relevant skills through natural language descriptions.

4.5.2 Three-Tier Quality Assurance

Level 1 (Automated): Every submission undergoes SDL schema validation, automated test execution against declared quality benchmarks, security scanning for unauthorized network access or data exfiltration, and performance profiling against latency and cost budgets.

Level 2 (Community): Published skills receive user ratings on five dimensions: accuracy, usefulness, ease of configuration, documentation quality, and value for price. Aggregated scores are displayed prominently in marketplace listings, and skills below minimum thresholds receive warning badges.

Level 3 (Curated): Safety-critical skills (medical, legal, financial, educational) undergo manual review by domain experts before receiving "Verified" status. Verified skills carry a trust badge and are prioritized in search results and recommendations.

4.5.3 Licensing and Monetization

The marketplace supports three licensing models: **Free** (open-source skills available to all users), **Freemium** (basic functionality free, advanced features paid), and **Premium** (paid skills with usage-based or subscription pricing). Skill developers set their own prices, with the platform charging a 20–30% commission. Revenue is distributed monthly via the developer's preferred payment method.

[**Figure 5** describes the Skill Marketplace Ecosystem. A circular diagram shows four participant types: Skill Developers (creating and publishing skills), Agent Builders (discovering and composing skills), End Users (interacting with composed agents), and the Platform (providing infrastructure, marketplace, and governance). Arrows show value flows: developers publish skills and receive revenue; builders compose agents and pay for skills; end users receive

AI services and provide usage data; the platform provides infrastructure and captures commissions.]

5. MODEL SELECTION AND COST ANALYSIS

5.1 Contemporary LLM Landscape

The selection of underlying AI models is among the most consequential architectural decisions for a skills-based agent platform. As of April 2026, the LLM landscape offers several model families, each with distinct trade-offs. This section provides a comprehensive analysis, with emphasis on the Anthropic Claude API as the recommended primary inference engine.[15]

5.2 Claude API Model Tiers

Anthropic offers three recommended model tiers:

Model	Input / 1M Tokens	Output / 1M Tokens	Context Window	Max Output	Key Strengths
Claude Opus 4.6	$5.00	$25.00	1,000,000	128,000	Complex reasoning, agentic tasks, extended thinking
Claude Sonnet 4.6	$3.00	$15.00	1,000,000	64,000	Best value, flagship quality, adaptive thinking
Claude Haiku 4.5	$1.00	$5.00	200,000	64,000	Speed, classification, routing, high-volume ops

Table 1. Claude API Model Tiers and Pricing (April 2026)

Opus 4.6 represents a 67% cost reduction over its predecessor Opus 4.1 ($15/$75), making frontier-class reasoning accessible for a much broader range of applications. It is the recommended model for skills requiring extended multi-step reasoning, complex analytical synthesis, or autonomous agentic tool use. Extended thinking tokens—internal reasoning generated before the final response—are billed at standard output rates ($25/1M).

Sonnet 4.6 is the recommended default for most SkillForge skills. It provides flagship-level performance at 40% lower cost than Opus, with identical 1M context window support. Sonnet

[15]Anthropic, "Claude API Pricing," platform.claude.com, April 2026.

also supports adaptive thinking, which automatically scales reasoning effort to match query complexity—simple requests skip expensive reasoning, while complex requests engage deep thinking. This naturally optimizes thinking token spend.

Haiku 4.5 is essential for SkillForge's infrastructure operations: intent classification, content routing, simple data extraction, and format validation. At $1/$5 per million tokens, it enables high-volume operations without significant cost impact.

5.3 Cost Optimization Strategies

Strategy	Mechanism	Typical Savings	Best For
Intelligent Model Routing	70/20/10 Haiku/Sonnet/Opus split	50–60%	Mixed-complexity workloads
Prompt Caching (5min TTL)	10% of input cost on cache hits	80–90% on input	Repeated system prompts
Prompt Caching (1hr TTL)	2x write, 10% read	70–85% on input	High-volume, stable prompts
Batch Processing	50% discount, 24hr window	50%	Non-real-time operations
Context Window Optimization	Pruning, summarization	20–40% on input	Long-context interactions
Response Caching	Direct cache serve	95–100%	Deterministic queries
Adaptive Thinking	Auto-scale reasoning tokens	30–50% on output	Variable-complexity skills

Table 2. Cost Optimization Strategies and Projected Savings

5.4 Cost Model: 1,000-Agent Deployment

The following model estimates monthly operating costs for a production SkillForge deployment serving 1,000 active agents, each processing an average of 100 interactions per day (3,000,000 total monthly interactions):

Cost Component	Monthly Volume	Rate	Gross Cost	After Optimization

Intent Classification (Haiku)	3M requests × 500 tokens	$1.00/1M in	$1,500	$150
Skill Execution: Sonnet	2.1M requests × 2K tokens	$3.00/1M in	$12,600	$2,520
Skill Execution: Opus	300K requests × 3K tokens	$5.00/1M in	$4,500	$1,350
Output Tokens (weighted)	3M requests × 800 tokens	~$12.00/1M out	$28,800	$8,640
Prompt Caching Credits	—	—	—	−6,480
Batch Processing Credits	—	—	—	−2,160
			Gross: $47,400	Net: $4,020

Table 3. Estimated Monthly API Cost Model for 1,000-Agent Deployment

Effective cost per interaction: $4,020 / 3,000,000 = $0.00134

With platform margin applied (3x markup for infrastructure, support, and margin), the end-user effective cost is approximately **$0.004 per interaction**—well within viable consumer and SME pricing models. This represents an optimization efficiency of **91.5%** compared to naive, unoptimized API usage.

5.5 Comparative Model Analysis

Dimension	Claude Sonnet 4.6	GPT-4o	Gemini 2.5 Pro	DeepSeek V3.2
Input Cost / 1M tokens	$3.00	$2.50	$1.25	$0.14
Output Cost / 1M tokens	$15.00	$10.00	$10.00	$0.28
Context Window	1,000,000	128,000	1,000,000	128,000
Max Output Tokens	64,000	16,384	65,536	8,192
Prompt Caching Discount	90%	50%	75%	90%
Extended Thinking	Yes (adaptive)	Yes (o-series)	Yes	Yes (R1 variant)
Arabic Language Quality	Strong	Good	Good	Limited
Agentic Tool Use	Excellent	Excellent	Good	Limited
Safety / Governance	Constitutional AI	RLHF	Standard	Minimal

Batch API	Yes (50% off)	Yes (50% off)	No	No
Data Residency Options	Yes (US, EU)	Yes	Yes	No

Table 4. Comparative LLM Analysis for Agent Platform Use (April 2026)

Recommendation: Claude Sonnet 4.6 as default, Haiku 4.5 for routing/classification, Opus 4.6 for complex reasoning. While DeepSeek V3.2 offers dramatically lower per-token pricing, its limited Arabic quality, minimal governance, constrained output length, and absence of batch API make it unsuitable as a primary engine. GPT-4o is a viable secondary provider for redundancy, but its smaller context window (128K vs 1M) limits applicability for knowledge-heavy skills.

6. MARKET AND COMPETITIVE ANALYSIS

6.1 Global AI Agent Market

The global AI agents market is experiencing extraordinary growth, with multiple research firms providing valuations that, while varying in methodology, converge on a consistent narrative of rapid expansion:[16]

Research Firm	2025 Value	2026 Projection	Long-Term Projection	CAGR
Grand View Research	$7.63B	$10.91B	$182.97B (2033)	49.6%
Fortune Business Insights	$8.03B	$11.78B	$251.38B (2034)	46.6%
Roots Analysis	—	$15.0B	$221.0B (2035)	34.6%
GM Insights	$5.9B	$7.7B	$105.6B (2034)	38.5%
SkyQuest	$40.95B*	$50.0B*	$247.01B (2033)	22.1%

Table 5. Global AI Agent Market Projections by Research Firm (*broader market definition)

Despite methodological differences, several structural findings are consistent:[17]

Enterprise Adoption: Approximately 51% of enterprises have AI agents in production as of 2026, with another 23% actively scaling. By year-end 2026, approximately 85% are expected to have implemented or planned deployments.

Return on Investment: Average ROI is $3.50 per $1 spent, with leading implementations achieving 8x returns. ROI compounds: 41% in year one, 87% in year two, 124%+ by year three.

Build vs. Buy: Ready-to-deploy agents hold 57–85% market share, but build-your-own (vertical) agents are growing at significantly higher CAGRs, reflecting increasing demand for customization—the precise segment SkillForge targets.[18]

[17]Ringly.io, "45 AI Agent Statistics You Need to Know in 2026," April 2026.
[18]Fortune Business Insights, "AI Agents Market Share, Size, Trends, Forecast, 2034," January 2026.

Governance Gap: Only 21% of companies have a mature agent governance model. Gartner projects that 40%+ of AI agent projects will be canceled by end of 2027 due to costs, unclear value, and weak governance. SkillForge's built-in governance framework addresses this directly.[19]

6.2 The Omani Strategic Opportunity

The Sultanate of Oman presents a compelling and strategically underserved market that uniquely aligns with SkillForge's value proposition.

6.2.1 National AI Programme (2024–2026)

Oman's National Programme for AI and Advanced Digital Technologies, running from 2024 through 2026, is structured around three pillars: (i) enhancing AI adoption in economic sectors, (ii) localizing AI technologies through public-private partnerships, and (iii) governing AI applications with a human-centered vision. The programme targets increasing the digital economy's GDP contribution from 2% (2021) to 5% (2030) to 10% (2040).[20]

SkillForge aligns directly with all three pillars. Its skills-based architecture enables AI adoption across diverse sectors without requiring each sector to develop its own AI expertise. Its marketplace model promotes localization by enabling Omani developers to create Arabic-first, culturally contextualized skills. Its governance framework supports human-centered AI with auditable agent behavior and configurable safety constraints.

6.2.2 The 2026–2030 Digital Economy Roadmap

In March 2026, the MTCIT unveiled a new five-year roadmap significantly expanding Oman's digital ambitions. Key elements include:[21]

[21] AI in Oman, "Oman Unveils 2026–2030 Digital Economy Roadmap," March 2026.

- A **national AI platform** for government use cases
- **Digital transformation centers** in all 11 governorates
- **Sovereign cloud infrastructure** with enhanced cybersecurity
- **Ma'een**, the first Omani national language model for Arabic NLP
- **RO 79 million** already invested in AI, producing 22 specialized companies
- **Digital economy** contributing **RO 800 million** to GDP (2023 baseline)

6.2.3 Digital Infrastructure Readiness

Metric	Value	Source
Digital Transformation Market (2025)	USD 2.72 billion	Mordor Intelligence
Projected Market (2030)	USD 4.65 billion (11.36% CAGR)	Mordor Intelligence
Internet Penetration	96.4% (4.44M users)	DataReportal 2023
Mobile Connections	6.29M (+2.61% YoY)	DataReportal 2023
Digital Social Inclusion Score	87.1%	Digital Economy Navigator 2025
Government Services Online (target)	80% by 2025	MTCIT
Subsea Cable Connectivity	50%+ of 21 GCC-connected cables	Otech Launch Report
AI Investment to Date	RO 79 million	MTCIT 2026 Report
IT Sector Omani Employees	1,289 across 33 professions	MTCIT 2026 Report

Table 6. Oman Digital Economy Key Metrics

The launch of **Omantel Otech** in February 2026—the first AWS-accredited sovereign cloud provider in the Middle East—provides the secure, compliant cloud infrastructure that enterprise AI agent deployment requires. Combined with the Muscat MC1 data center's connectivity to

120+ cities globally, Oman's position creates favorable conditions for low-latency AI services across the Gulf.[22]

6.3 Competitive Differentiation

Dimension	SkillForge	ChatGPT / Claude	Copilot Studio	Agentforce	LangChain
Paradigm	Skill composition	Prompt config	Low-code flows	CRM-specific	Developer framework
Technical Barrier	No-code	Low (text)	Low-code	Admin-level	High (Python)
Modularity	Atomic skills	Monolithic	Actions/flows	CRM objects	Chains/agents
Skill Reusability	Full marketplace	None	Templates only	Limited	GitHub repos
Model Flexibility	Multi-model routing	Single vendor	Microsoft only	Salesforce AI	Multi-model
Arabic First-Class	Yes	Secondary	Limited	Minimal	No
Pricing Model	Per-skill usage	Subscription	Per-seat license	Enterprise	Free + hosting
Ecosystem Lock-In	Open SDL standard	Vendor locked	Microsoft 365	Salesforce	Python ecosystem
Governance	Skill-level audit	Conversation-level	Enterprise IAM	Enterprise	Custom
Quality Assurance	3-tier automated	None	Manual testing	Internal QA	Manual
Marketplace Economics	Two-sided market	GPT Store (basic)	None	AppExchange	None

Table 7. Comprehensive Competitive Differentiation Matrix

[22]AI in Oman, "Omantel Launches Otech," February 2026.

7. IMPLEMENTATION METHODOLOGY

7.1 Technology Stack Selection

The SkillForge platform is implemented using a cloud-native technology stack optimized for scalability, developer experience, and the specific requirements of AI agent orchestration:

Frontend Layer: Next.js 14 with React Server Components provides the agent builder interface. Tailwind CSS and shadcn/ui deliver a polished, accessible design system. The composition interface employs a drag-and-drop paradigm using a reactive state management architecture, with real-time preview of agent behavior as skills are added or configured. The frontend is fully bilingual (English/Arabic) with RTL layout support as a first-class concern.

API Layer: A Node.js/Express API gateway handles authentication (OAuth 2.0 with PKCE), rate limiting, request validation, and routing. The gateway implements the Backend-for-Frontend (BFF) pattern, providing optimized endpoints for the web UI, mobile app, WhatsApp integration, and third-party API consumers.

Core Services: The ACE is implemented as a set of microservices communicating through an event bus (NATS or Redis Streams). Services include: Composition Service (dependency resolution, validation), Routing Service (model selection, cost optimization), Execution Service (model invocation, response synthesis), Cache Service (multi-tier caching management), and Telemetry Service (metrics, logging, billing).

Data Layer: Supabase (PostgreSQL) serves as the primary database for relational data. The pgvector extension enables semantic search over skill descriptions and user queries. Redis provides session management, real-time caching, and pub/sub messaging. Object storage (S3-compatible) stores skill packages, user-uploaded knowledge files, and response artifacts.

AI Integration Layer: An abstraction layer wraps the Anthropic Claude API (primary), with extension points for OpenAI, Google Vertex AI, and other providers. The abstraction normalizes request/response formats, implements retry logic with exponential backoff, manages API key rotation, and provides unified metrics across providers.

Delivery Channels: WhatsApp Business API integration targets the Omani market (90%+ WhatsApp penetration). Additional channels include web embed (iframe), REST API (for developer integrations), and native mobile SDKs (React Native/Expo).

7.2 Development Phases

Phase 1: Foundation (Months 1–3). Core skill execution engine with basic SDL schema. Five foundational skills: Text Generation, Document Creation (DOCX/PDF), Web Research, Data Analysis, and Arabic Language Enhancement. Minimal web interface for agent composition with drag-and-drop skill selection. Single-model support (Claude Sonnet 4.6). Internal testing with 20–50 beta users.

Phase 2: Platform (Months 4–6). Full ACE implementation with dependency resolution, context bus, and model routing (Haiku/Sonnet/Opus). Prompt caching integration achieving 80%+ cache hit rates. User authentication, multi-tenancy, and role-based access control. WhatsApp delivery channel for Omani market. Expanded skill library to 20–30 skills. Public beta with 200–500 users.

Phase 3: Marketplace (Months 7–12). Skill Marketplace with publication, discovery, licensing, and payment workflows. Three-tier quality assurance system (automated, community, curated). SDK and documentation for third-party skill developers. Analytics dashboard for skill

developers and agent builders. Enterprise features (SSO, audit logging). Target: 1,000+ users, 100+ marketplace skills.

Phase 4: Scale (Months 13–18). Multi-agent orchestration capabilities. Arabic-first skill curation programme with Omani domain experts. GCC market expansion (UAE, Saudi Arabia, Bahrain, Kuwait, Qatar). Advanced governance features (compliance templates, audit trails, data residency). Strategic partnerships with government and enterprise clients. Target: 10,000+ users, 500+ skills, GCC presence.

7.3 Quality Assurance Strategy

AI agent quality assurance is fundamentally different from traditional software testing due to the probabilistic nature of LLM outputs. SkillForge implements a multi-layered testing strategy:

Deterministic Testing: Verifies that skill compositions resolve correctly, context propagation follows declared schemas, tool integrations return expected responses (using mocks), and SDL validation catches schema violations. These tests achieve 100% pass/fail determinism and run on every commit.

Probabilistic Testing: Evaluates agent response quality across standardized test suites using statistical acceptance criteria. Each test case specifies expected behavior with tolerance ranges (e.g., “accuracy > 0.85 on medical terminology questions”). Tests are run N times (typically N=50) and results are compared against statistical thresholds using the binomial test for pass rates and the Mann-Whitney U test for quality score distributions.

Adversarial Testing: Probes agent behavior under adversarial conditions including prompt injection attempts, malicious tool use, conflicting skill instructions, and context overflow.

Adversarial test suites are maintained by a dedicated security team and updated in response to emerging attack vectors.

Human-in-the-Loop Evaluation: Beta users evaluate agent behavior in real-world scenarios using structured feedback forms. Feedback is aggregated into skill quality scores and used to identify systematic failure modes that automated testing may miss.

8. BUSINESS MODEL AND UNIT ECONOMICS

8.1 Revenue Streams

SkillForge implements a diversified revenue model capturing value at each platform layer:

Stream 1 — Platform Subscription: Tiered plans ranging from Free (3 skills, 50 interactions/day) through Professional ($29/month, 15 skills, 500 interactions/day) to Enterprise (custom pricing, unlimited skills, SLA guarantees, dedicated support). Annual billing offers 20% discount.

Stream 2 — Marketplace Commission: 25% commission on premium skill sales. Skill developers retain 75% of revenue. Commission rate decreases to 20% for high-volume developers (>$10,000/month revenue) to incentivize quality and scale.

Stream 3 — API Usage Margin: Pass-through AI model costs with 3x markup that covers infrastructure, caching optimization, support, and margin. The markup is justified by the 85–92% cost reduction achieved through intelligent routing and caching.

Stream 4 — Enterprise Services: Custom deployment, dedicated infrastructure, SLA guarantees, priority support, compliance consulting, and custom skill development for large organizations.

Stream 5 — Skill Development Services: Professional services for organizations requiring custom skill creation, including domain knowledge encoding, quality assurance, and ongoing maintenance. Priced at project-based rates.

8.2 Unit Economics and Projections

Metric	Year 1	Year 2	Year 3
Active Users	2,000	15,000	75,000
Marketplace Skills	150	800	3,000
Avg Interactions/User/Day	30	50	75
Platform Subscription Revenue	$180K	$1.8M	$12M
Marketplace Commission Revenue	$30K	$400K	$3M
API Margin Revenue	$120K	$1.5M	$9M
Enterprise Services Revenue	$50K	$500K	$4M
Total Revenue	$380K	$4.2M	$28M
AI API Costs	$96K	$960K	$5.4M
Infrastructure Costs	$48K	$240K	$720K
Team Costs	$360K	$1.2M	$3.6M
Gross Margin	-32%	43%	65%

Table 11. Revenue Model Projections (Year 1–3)

8.3 Omani Market Pricing

The Omani market requires pricing sensitivity. SkillForge implements **regional pricing** with 30–40% discount for users in Oman, partially offset by lower support costs (Arabic support is native, not an add-on) and government partnership revenue. The Omani government's stated commitment to supporting AI-focused tech startups and its RO 79 million AI investment to date create potential for grant funding, subsidized licenses, and public-private partnerships.

9. RISKS, LIMITATIONS, AND ETHICAL CONSIDERATIONS

9.1 Risk Assessment Matrix

Risk Category	Risk	Probability	Impact	Mitigation Strategy
Technical	Model API deprecation/pricing change	Medium	High	Multi-model abstraction; negotiated SLAs; contingency routing
Technical	Skill quality variance at scale	High	Medium	Three-tier QA; automated testing; delisting policy
Technical	Context window exhaustion	Medium	Medium	Progressive pruning; hierarchical summarization; skill splitting
Technical	Prompt injection via malicious skills	Medium	High	SDL security scanning; sandboxed tool execution; input sanitization
Market	Competition from well-funded incumbents	High	High	Niche focus (Arabic-first); skills moat; marketplace network effects
Market	Slow user adoption/education barrier	Medium	High	Freemium model; template agents; partnership with gov programs
Market	AI project cancellation trend (Gartner: 40%+)	Medium	Medium	Clear ROI metrics; low entry cost; rapid time-to-value
Regulatory	Evolving Omani data privacy framework	Medium	Medium	Privacy-by-design; data residency; sovereign cloud options
Ethical	Bias propagation through skills	Medium	High	Bias testing in QA; diverse skill developer program; reporting mechanisms
Ethical	Safety-critical domain errors	Low	Very High	Mandatory disclaimers; HITL requirements; verified skill program

Table 10. Risk Assessment Matrix

9.2 Ethical Framework

SkillForge adopts a **responsible AI framework** grounded in four principles:

Transparency: All agent outputs are clearly attributed as AI-generated. Skill compositions are inspectable—users can see which skills contributed to any response. Governance logs maintain full audit trails of agent decisions.

Safety: Skills in regulated domains (medical, legal, financial) carry mandatory safety classifications and use disclaimers. High-stakes decisions require human-in-the-loop confirmation. The platform implements Anthropic's Constitutional AI principles through its model selection and skill governance layer.

Fairness: The three-tier QA process includes bias testing. The skill marketplace actively recruits diverse skill developers to ensure that skills reflect varied perspectives and cultural contexts. Arabic-language skills are developed with sensitivity to Gulf cultural norms and Omani social conventions.

Privacy: Strict data isolation between skills prevents cross-context data leakage. User data is processed in compliance with applicable regulations. Sovereign cloud deployment options ensure data residency requirements are met.

10. FUTURE RESEARCH DIRECTIONS

10.1 Autonomous Skill Generation

A natural evolution is the ability for the platform to generate new skills autonomously from user behavior patterns. Research should explore how foundation models can synthesize SDL definitions from examples of desired behavior, enabling a self-improving skill ecosystem. Preliminary experiments suggest that Claude Opus 4.6 with extended thinking can generate valid SDL schemas from 10–15 examples of desired agent behavior, though quality and reliability require further investigation.

10.2 Multi-Agent Orchestration

Current architecture focuses on single-agent, multi-skill composition. Future work should explore multi-agent architectures where specialized SkillForge agents collaborate on complex tasks. This aligns with industry trends: multi-agent systems are projected to grow at the highest CAGR within the AI agent market, as enterprises discover that complex workflows benefit from agent specialization and collaboration rather than monolithic capability.[23]

10.3 Arabic Language Model Optimization

While Claude provides strong Arabic support, significant room exists for improvement in Gulf dialect understanding, Omani cultural context modeling, and Arabic-specific evaluation benchmarks. Future research should explore: (a) integration with Oman's Ma'een language model for government-specific use cases, (b) fine-tuning strategies for Gulf Arabic dialect recognition, and (c) development of Arabic-specific agent quality benchmarks that evaluate cultural appropriateness alongside linguistic accuracy.[24]

[23]Roots Analysis, "AI Agents Market," January 2026. Projects $221B by 2035.
[24]UNESCO, "Sultanate of Oman: AI Readiness Assessment Report," 2025.

10.4 Federated Skill Learning

Privacy-preserving skill improvement through federated learning represents a promising direction. Skills could improve based on aggregated usage patterns across the platform without exposing individual user data, creating a virtuous cycle where quality improves with scale while maintaining data sovereignty—a critical requirement for government and enterprise clients in the GCC.

10.5 Quantum-Ready Architecture

Oman's National Programme includes quantum computing initiatives. Future research should explore quantum computing integration for optimization-heavy skills such as logistics planning, financial portfolio optimization, and drug discovery compound screening. While quantum advantage for these applications remains speculative, architectural preparation ensures SkillForge can incorporate quantum capabilities as they mature.

10.6 Skill Interoperability Standards

As the AI agent ecosystem matures, interoperability between platforms will become increasingly important. SkillForge's SDL could evolve into an open standard for AI skill specification, analogous to how Docker's container format became an industry standard. Research should explore formal standardization processes, cross-platform skill portability, and federation between SkillForge instances operated by different organizations.

11. CONCLUSION

This paper has presented SkillForge, a comprehensive architecture for democratizing AI agent construction through a modular, skills-based paradigm. The platform addresses a fundamental gap in the current AI landscape: the disconnect between the extraordinary capability of modern foundation models and the primitive mechanisms available to most users for harnessing that capability.

The skills-based approach—grounded in established principles of component-based software engineering (Szyperski, 2002), distributed cognition theory (Hutchins, 1995; Clark & Chalmers, 1998), and platform economics (Rochet & Tirole, 2003; Parker et al., 2016)—offers a theoretically sound and commercially viable pathway to making AI agent construction accessible to the hundreds of millions of potential users who currently lack the technical expertise to build custom agents.

The **Skill Definition Language** provides a formal specification for encoding domain expertise into reusable, composable modules with typed interfaces, quality guarantees, and versioned evolution. The **Agent Composition Engine** solves the complex orchestration problems that arise when multiple skills must collaborate—including dependency resolution, context propagation, and model routing. The **Runtime Execution Framework** delivers production-grade performance through a four-tier caching architecture, intelligent model routing, and serverless scalability. The **Skill Marketplace** creates the economic incentives necessary to build a thriving two-sided market of skill developers and agent builders.

Our cost analysis demonstrates that, with aggressive optimization, a skills-based platform can achieve per-interaction costs of approximately **$0.004** (with platform margin)—a level that supports viable consumer and SME pricing. The Claude API, particularly Claude Sonnet 4.6 at

$3/$15 per million tokens, provides the optimal foundation for this platform, offering the best balance of capability, cost, governance, and Arabic language support.

The market opportunity is substantial: the global AI agent market is projected to exceed $180 billion by 2033, with build-your-own segments growing at the highest rates. The Sultanate of Oman, with its National AI Programme, 2026–2030 Digital Economy Roadmap, RO 79 million in AI investment, and Otech sovereign cloud infrastructure, presents a compelling entry market with natural expansion paths across the GCC.

SkillForge is not merely a product concept but a thesis about the future of human-AI interaction. We posit that the next decade will see a fundamental shift from AI as a monolithic service to AI as a composable ecosystem, where the ability to assemble, configure, and deploy purpose-built agents becomes as natural and accessible as assembling a playlist or customizing a smartphone. The skills-based paradigm—with its roots in proven software engineering principles, its alignment with cognitive science theory, and its commercial viability through platform economics—represents the most promising architecture for enabling this shift.

The question is not whether AI agents will become ubiquitous—the market data makes that conclusion inescapable. The question is who will build them, who will benefit from them, and how accessible the construction process will be. SkillForge's answer is clear: **everyone should be an agent builder.**

REFERENCES

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., and Polosukhin, I. (2017). "Attention Is All You Need." Advances in Neural Information Processing Systems, 30, 5998–6008.

[2] Brown, T., Mann, B., Ryder, N., et al. (2020). "Language Models are Few-Shot Learners." Advances in Neural Information Processing Systems, 33, 1877–1901.

[3] Wei, J., Wang, X., Schuurmans, D., et al. (2022). "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models." Advances in Neural Information Processing Systems, 35.

[4] Yao, S., Zhao, J., Yu, D., et al. (2023). "ReAct: Synergizing Reasoning and Acting in Language Models." International Conference on Learning Representations (ICLR).

[5] Schick, T., Dwivedi-Yu, J., Dessì, R., et al. (2023). "Toolformer: Language Models Can Teach Themselves to Use Tools." Advances in Neural Information Processing Systems, 36.

[6] Szyperski, C. (2002). Component Software: Beyond Object-Oriented Programming, 2nd ed. Addison-Wesley Professional.

[7] Hutchins, E. (1995). Cognition in the Wild. MIT Press.

[8] Clark, A. and Chalmers, D.J. (1998). "The Extended Mind." Analysis, 58(1), 7–19.

[9] Rochet, J.C. and Tirole, J. (2003). "Platform Competition in Two-Sided Markets." Journal of the European Economic Association, 1(4), 990–1029.

[10] Parker, G.G., Van Alstyne, M.W., and Choudary, S.P. (2016). Platform Revolution: How Networked Markets Are Transforming the Economy. W.W. Norton & Company.

[11] Grand View Research. (2026). "AI Agents Market Size and Share: Industry Report, 2033." San Francisco, CA.

[12] Fortune Business Insights. (2026). "AI Agents Market Share, Size, Trends, Forecast, 2034." Pune, India.

[13] Roots Analysis. (2026). "AI Agents Market: Global Market Size, Share, Trends, 2035." New Delhi, India.

[14] DemandSage. (2026). "AI Agents Market Size, Share & Trends: 2026–2034 Data."

[15] GM Insights. (2025). "AI Agents Market Size, Share, and Growth Analysis."

[16] SkyQuest Technology. (2026). "AI Agents Market Size, Share, and Growth Analysis."

[17] MarketsandMarkets. (2026). "Agentic AI Market Report 2025–2032."

[18] Ministry of Transport, Communications and Information Technology. (2024). "Oman AI & Digital Future Program (2024–2026)." Muscat, Sultanate of Oman.

[19] AI in Oman. (2026). "Oman Unveils 2026–2030 Digital Economy Roadmap: National AI Platform, Digital Centres, and a Path to 10% of GDP." March 22, 2026.

[20] AI in Oman. (2026). "Omantel Launches Otech: A Game Changer for Digital Sovereignty in the Middle East." February 15, 2026.

[21] UNESCO. (2025). "Sultanate of Oman: Artificial Intelligence Readiness Assessment Report."

[22] Oxford Business Group. (2025). "Boost to Oman's Artificial Intelligence Drives Economic Diversification." The Report: Oman 2025, Chapter: ICT.

[23] Mordor Intelligence. (2026). "Oman Digital Transformation Market Size & Share Analysis."

[24] Anthropic. (2026). "Claude API Pricing." platform.claude.com/docs/en/about-claude/pricing. Retrieved April 2026.

[25] Gartner. (2026). "Top Strategic Technology Trends for 2026." Stamford, CT.

[26] Deloitte. (2026). "State of Generative AI in the Enterprise: Q1 2026 Report."

[27] Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System." (Structural reference for whitepaper format).

[28] Anthropic. (2024). "Constitutional AI: Harmlessness from AI Feedback." Technical Report.

[29] SAMENA Council. (2024). "Oman Launches National Programme for AI and Digital Technologies."

[30] Enterprise AI Expo. (2026). "The Middle East’s Definitive Platform for Enterprise-Grade AI." Muscat, Oman.

APPENDIX A: GLOSSARY OF TECHNICAL TERMS

Term	Definition
ACE	Agent Composition Engine — runtime orchestrator for multi-skill agents within SkillForge
Agentic AI	AI systems capable of autonomous reasoning, planning, tool use, and multi-step task execution
CAGR	Compound Annual Growth Rate — annualized growth rate over a specified time period
Constitutional AI	Anthropic's approach to AI alignment using AI-generated feedback against principles
DAG	Directed Acyclic Graph — data structure used for skill dependency resolution
Extended Thinking	LLM feature generating internal reasoning tokens before final response
LLM	Large Language Model — transformer-based models (e.g., Claude, GPT, Gemini)
LoRA/QLoRA	Low-Rank Adaptation — efficient fine-tuning technique for large models
RAG	Retrieval-Augmented Generation — technique injecting external knowledge via vector search
REF	Runtime Execution Framework — serverless infrastructure for agent deployment
SDL	Skill Definition Language — formal JSON-Schema specification for encoding skill modules
SMP	Skill Marketplace Protocol — standards for skill publishing, discovery, and distribution
Semantic Versioning	Version numbering scheme (MAJOR.MINOR.PATCH) for managing compatibility
TTL	Time to Live — duration for which cached data remains valid before expiration
Two-Sided Market	Platform connecting two distinct user groups who benefit from each other's participation

Table A.1. Glossary of Technical Terms

APPENDIX B: SDL SCHEMA SPECIFICATION (ABRIDGED)

The following represents an abridged JSON Schema specification for a complete Skill Definition:

```
{
  "$schema": "https://skillforge.nuqtai.com/sdl/v1.0/schema.json",
  "skillDefinition": {
    "metadata": {
      "id": "uuid-v4",
      "name": "Medical Research Assistant",
      "version": "1.2.0",
      "author": { "id": "...", "name": "...", "org": "..." },
      "license": "MIT",
      "tags": ["medical", "research", "pubmed", "evidence-synthesis"],
      "description": {
        "short": "AI-powered medical literature review and evidence synthesis",
        "long": "Comprehensive medical research skill that searches PubMed..."
      },
      "compatibility": {
        "models": ["claude-opus-4-6", "claude-sonnet-4-6"],
        "languages": ["en", "ar", "ar-OM"],
        "runtimeVersion": ">=1.0.0"
      }
    },
    "capabilities": {
      "inputs": [
        { "name": "research_question", "type": "string", "required": true },
        { "name": "date_range", "type": "object", "required": false },
        { "name": "inclusion_criteria", "type": "array", "required": false }
      ],
      "outputs": [
        { "name": "literature_summary", "type": "structured", "format": "json" },
        { "name": "evidence_quality", "type": "string", "format": "grade" },
        { "name": "citations", "type": "array", "format": "apa7" }
      ]
    },
    "knowledge": {
      "systemPrompt": "You are a medical research assistant...",
      "ragConfig": {
        "vectorStore": "https://vectors.skillforge.com/pubmed-2026",
        "embeddingModel": "text-embedding-3-large",
        "topK": 20,
        "similarityThreshold": 0.78
      }
    },
    "tools": [
      {
        "name": "pubmed_search",
        "type": "api",
        "endpoint": "https://eutils.ncbi.nlm.nih.gov/entrez/...",
        "rateLimit": { "requests": 10, "period": "1s" }
      }
    ],
    "quality": {
```

```
        "benchmarks": [
          { "name": "medical_accuracy", "minScore": 0.90 },
          { "name": "citation_correctness", "minScore": 0.95 }
        ]
      }
    }
  }
```

APPENDIX C: EXECUTION PIPELINE DETAILED SEQUENCE

The following describes the complete execution sequence for a user interaction with a SkillForge agent, including timing targets and failure modes:

Step 1 — User Input Reception *[Target: <10ms]*

Platform receives message via web/API/WhatsApp. Input is sanitized, rate-limited, and logged. Session context is loaded from persistent store.

Step 2 — Intent Classification *[Target: <50ms]*

Haiku 4.5 classifies intent into skill categories. Classification confidence score determines whether single or multi-skill execution is triggered. Confidence <0.6 triggers clarification request.

Step 3 — Skill Selection & DAG Construction *[Target: <20ms]*

ACE selects optimal skill subset. Dependency DAG is constructed via topological sort. Parallel execution groups are identified for independent skills.

Step 4 — Context Assembly *[Target: <30ms]*

Context bus aggregates: conversation history (last N turns), user preferences, skill system prompts, RAG retrieval results, tool credentials. Total context is measured against model window limits; pruning triggered if >80% of window.

Step 5 — Model Routing *[Target: <10ms]*

Each skill is assigned to optimal model based on capability matrix, user's cost-quality preference, and current model availability. Cache-hit check performed for prompt caching.

Step 6 — Skill Execution *[Target: 200ms–5s]*

Skills execute against assigned models. Independent skills run in parallel. Dependent skills execute sequentially per DAG. Extended thinking enabled for Opus-routed skills. Tool invocations are sandboxed with timeout enforcement.

Step 7 — Response Synthesis *[Target: 100ms–1s]*

Sonnet 4.6 synthesizes individual skill outputs into coherent response. Conflict resolution applied for contradictory outputs. Format compliance validated against output schema.

Step 8 — Quality Validation *[Target: <100ms]*

Response checked against active skills' quality metrics. Safety classifier verifies absence of harmful content. Format compliance verified. Failures trigger: (a) retry with escalated model, (b) partial response with disclaimer, or (c) human-in-the-loop escalation.

Step 9 — Caching & Telemetry *[Target: <20ms]*

Deterministic outputs cached. Prompt cache updated. Interaction metrics recorded: total latency, per-skill latency, model tokens consumed, cache hit rate, quality scores. Billing event emitted.

Step 10 — Response Delivery *[Target: <50ms]*

Validated response delivered via originating channel. Streaming enabled for web/API channels. WhatsApp messages chunked for readability. Delivery confirmation logged.