UDC 004.8:336.77
DOI: https://doi.org/ 10.17721/3041-2323.2025.361-376

Ramin SAMADOV, PhD in Technical Sciences
ORCID ID: 0009-0002-9384-6541
e-mail: ramin.samedov@gmail.com
Baku State University, Baku, Azerbaijan

ARCHITECTURE AND INTERACTION ALGORITHMS OF BANKING AND PAYMENT SYSTEMS IN THE AUTOMATED CREDIT LIMIT ESTABLISHMENT PROCESS

The automation of credit limit establishment represents a critical intersection of risk management, financial technology, and economic policy. This article examines the architectural frameworks and algorithmic interactions between banking and payment systems that enable automated credit decision-making. Based on systematic analysis of contemporary banking infrastructures, we identify key components including scoring engines, processing centers, and card management platforms that interact through REST APIs and asynchronous messaging systems (Apache Kafka). The research reveals that modern credit limit automation systems employ hybrid architectures combining microservices principles with event-driven communication patterns, enabling real-time risk assessment while maintaining regulatory compliance. Findings demonstrate that implementation of such systems directly impacts financial inclusion, credit portfolio quality, and operational efficiency of financial institutions.

Keywords: ***credit scoring, banking architecture, risk management, financial technology, API integration, payment systems, microservices, real-time processing.***

Introduction

Retail lending serves as one of the key drivers of economic growth and satisfaction of consumer needs in the modern financial system. Credit cards, being a central product in this segment, provide customers with convenient and operational access to borrowed funds. However, the provision of credit limits is associated with significant risks for credit organizations related to potential borrower default. In this regard, the process of establishing credit limits represents a critically important stage in the customer service cycle, intersecting risk management, data analysis, and customer-oriented service.

The digital transformation of the banking industry has generated new challenges and opportunities despite traditional scoring methods and creditworthiness assessment approaches being well-studied in economic literature. The implementation of new banking systems, credit product management systems (such as Card Credit Management Platform – CCMP), and their necessary integration into existing technological landscapes require deep rethinking of decision-making process architecture. The relevant task becomes not only the development of predictive models based on machine learning but also the creation of reliable, secure, and efficient mechanisms for interaction between heterogeneous systems (such as processing centers, scoring engines, and business process automation platforms) (Smith, 2024).

This research aims to analyze and improve the process of establishing credit limits within the framework of credit card issuance based on a comprehensive approach combining risk assessment methodology and modern integration solutions. The object of the study is the retail lending process in commercial banks (Johnson, 2024). The subject of the study is the process of establishing credit limits and its organizational-technological support. The work applies methods of system analysis, business process modeling, analysis of modern IT solutions in banking integration (microservice architecture, REST API), and analysis of financial models for assessing credit risks.

Table 1 summarizes the key challenges associated with automated credit limit establishment.

Table 1

Key Challenges in Automated Credit Limit Establishment

Challenge Category	Specific Challenges	Impact on System Architecture
Data Integration	Heterogeneous data sources, real-time processing requirements	Necessitates robust API governance and message brokering
Risk Modeling	Predictive accuracy, model interpretability, regulatory compliance	Requires specialized scoring engines and validation frameworks

Continuation of Table 1

Challenge Category	Specific Challenges	Impact on System Architecture
System Performance	High availability, low latency, scalability	Demands microservices architecture and cloud-native solutions
Security Concerns	Data protection, fraud prevention, secure authentication	Implements encryption, IAM systems, and continuous monitoring

Methodology

This research employs a comprehensive methodology combining systematic analysis of architectural patterns, examination of integration technologies, and assessment of risk assessment frameworks. The methodological approach includes:

– Systematic Analysis: Examination of existing banking architectures through analysis of integration patterns, component relationships, and data flow mechanisms;

– Business Process Modeling: Mapping of credit limit establishment processes using BPMN notation to identify critical interaction points and potential bottlenecks;

– Technology Assessment: Evaluation of contemporary IT solutions in banking integration including microservices architecture, REST API implementations, and message brokering systems;

– Risk Model Analysis: Review of financial risk assessment models and their integration points within technological infrastructures.The research methodology follows a structured approach to understanding the complex interactions between organizational requirements, technological capabilities, and regulatory constraints in automated credit limit establishment (Anderson, 2023).

Results

The automated credit limit establishment process relies on a sophisticated architectural framework that integrates multiple banking systems through standardized protocols and interfaces. Based on our analysis, the optimal architecture represents a hybrid approach

combining microservices principles with event-driven communication patterns.

The credit limit automation ecosystem consists of several critical technical components that ensure stable operation, data processing, and user interaction:

– Application Server: Hosts the core business logic of the system, processes user requests, executes operations, and interacts with databases while supporting scalability and fault tolerance. Recommended technologies include Spring Boot framework with application server options (Tomcat, Jetty, or Undertow);

– Database: Stores information about customers, credit products, limits, debts, and transactions while ensuring data integrity and high query processing speed with support for replication and backup mechanisms. Oracle DB represents a suitable technological solution for this component;

– Integration Layer: Responsible for interaction with other banking systems through API-based data transmission. This layer typically employs Message Broker systems (Apache Kafka, RabbitMQ) and HTTP protocols for synchronous communication;

– Authentication and Authorization: Ensures user authentication and authorization while implementing encryption mechanisms and access control. Recommended solutions include Keycloak or Spring Security with OAuth2/JWT tokens;

– Cache Storage: Provides rapid access to frequently used data through implementation of Redis or similar in-memory data storage systems.

The end-user functionality is delivered through multiple application interfaces:

– Web Application for Bank Employees: An interface for managing credit card products, monitoring debts, and processing customer requests. This interface should be implemented within the existing banking system used by back-office personnel. React framework provides suitable technological foundation for this interface;

– Mobile Application for Customers: Allows customers to view credit limits, debts, payment schedules, and perform payments. This functionality should be implemented within the existing banking

application used by customers. Recommended technologies include Flutter, React Native, Kotlin, or Swift depending on existing infrastructure and development capabilities;

– API Interfaces (REST/GraphQL/SOAP): Ensures interaction with other banking systems and supports integration with external services. REST API architecture represents the current industry standard for such integrations.

Security and Monitoring Mechanisms

The operational integrity of credit limit systems requires robust security and monitoring implementations:

– Monitoring System: Tracks the status of servers, databases, and applications using tools such as Prometheus, Grafana, or Zabbix;

– Logging System: Records system events for analysis and diagnostics through implementation of Elasticsearch, Logstash, and Kibana (ELK stack);

– Access Management System: Provides flexible configuration of user roles and access rights with integration to Active Directory or other authentication systems.

Algorithmic Interactions in Credit Limit Establishment

The process of card issuance and credit limit establishment consists of multiple stages requiring precise algorithmic interactions between systems. The credit limit establishment process begins when a customer submits an application through available channels (mobile application, web version, or bank branch). The algorithmic workflow follows these stages:

Product Selection: In the mobile application or website, the customer is provided with the opportunity to choose a card with indication of credit limit conditions such as limit validity period, plastic type, minimum and maximum credit limit amount, grace period duration, and interest rate. The list of available products is displayed in the interface through calling a service of actual products from systems via REST API, ensuring relevance and accuracy of information presented to the customer when choosing a credit product (Williams, 2023).

Customer Data Input and Application Submission: The customer enters identification data including FIN code and mobile phone number and provides consent to receive data from Asan Finance/AKB

systems. Signing of consent and confirmation of credit card application is performed using a one-time password (OTP), guaranteeing security and confirmation of customer action.

Credit Limit Decision Making: At the next stage of the process, internal bank systems make the decision to provide or decline issuance of a credit card with limit. For decision making, necessary data is requested from ASAN Finance and AKB systems. After the bank's positive decision, the customer is provided with an offer with card conditions, which they must either accept or decline.

Card Agreement Processing

Upon customer acceptance and signing of the agreement, the system processes the request through the following algorithmic steps:

– A request to create or edit the customer's card is sent to the processing center system depending on whether the customer is new or existing, accomplished through web services of processing;

– Information about creation (issuance) of the credit card, including limit establishment, is transmitted through web services;

– The request to create or edit the customer's card is also sent to the CCMP system through REST API, where the card agreement with established limit is registered;

– The customer's card is then opened in the banking system, appropriate accounts are created, and data about the limit is updated, ensuring information relevance for further servicing and transaction processing.

The Algorithm of Automated Credit Limit Establishment Process for customer can be showed as follow:

Step 1. Application Submission.

The algorithm is initiated by a client submitting a request for a credit card. The submission can occur through multiple available channels:

– Channel 1: Mobile application.
– Channel 2: Web portal.
– Channel 3: Physical bank branch.

Step 2. Product Selection

For digital channels (Mobile App/Web):

– The interface displays available credit card products by invoking a REST API service to fetch the current list from the bank's product catalog.

– Displayed product details include credit limit conditions (validity period, min/max amount), card type (plastic), grace period duration, and interest rates.

– For the branch channel:

– A bank employee assists the client and manually initiates the product selection request through the Back-Office System interface.

This step ensures the client receives accurate and up-to-date product information for an informed selection.

Step 3. Client Data Input and Application Finalization

1. The client provides required identification data, including:

– FIN (Personal Identification Number).

– Mobile phone number.

2. The client provides explicit digital consent for the bank to retrieve their credit data from external bureaus (Asan Finance and AKB).

3. To securely confirm the application and consent, the client authenticates the action using a One-Time Password (OTP) sent to their registered mobile number.

Step 4. Credit Limit Approval Decision

The bank's internal scoring system requests the client's credit data from the external systems (Asan Finance and AKB).

Based on the retrieved data and internal risk models, the system renders a decision:

Condition 4.1: Decision Approved. The system generates a formal offer (officer) detailing the card's terms and conditions. This offer is presented to the client for their acceptance or rejection.

Condition 4.2: Decision Declined. The process terminates, and the client is notified of the rejection.

Step 5. Contract Formalization and Card Issuance

This step executes only if the client accepts the offer (Condition 4.1).

1. Processing Center Request: A request to create (for new clients) or edit (for existing clients) a card record is sent to the Processing Center via secure web services.

2. CCMP Registration: A simultaneous request is sent to the Card Management System (CCMP) via REST API to formally register the card agreement and the established credit limit.

3. Core Banking Setup: Upon successful processing:

– The card is officially opened in the core banking system.

– Necessary financial accounts are created.

– The credit limit is updated across all relevant systems, enabling future transactions.

Risk Assessment Integration

The critical algorithmic component of credit limit establishment involves integration with risk assessment systems:

– Scoring Engine Interaction: The CCMP system communicates with external scoring engines through API Gateway, which sends requests to AKB and Asan Finance via their SOAP or REST APIs.

– Predictive Analytics: The scoring engine (which may be a module within CCMP or a separate microservice) executes predictive models (logistic regression, gradient boosting) producing outputs including score (e.g., 720), decision (APPROVE/DECLINE) and recommended_limit.

– Limit Calculation: The CCMP system establishes the final limit based on the recommended limit and product rules (min, max limit). The application status is set to APPROVED.

– Event Notification: The CCMP system through Kafka sends a CreditLimitApproved event with customer and limit data. This event can be consumed by other systems (e.g., CRM for marketing, Analytical storage).

Operational Processing of Credit Card Transactions

The operational process of conducting credit card transactions involves all key stages necessary for successful transfer of funds from the customer's credit account to the merchant's account or for cash withdrawal. The process of transaction operation using a credit card includes several key steps that guarantee correctness, security, and transaction completion. A customer performs an operation using a credit card which may include payment for goods in partner and non-partner networks through POS-terminals, cash withdrawal at ATMs, or execution of P2P-transfer from a credit card (Richardson, & Smith, 2019).

Data about the authorized transaction is directed to the processing system through the central Bank's system where verification and authorization are performed. Then the card system, depending on configured tariffs and operation type, determines the commission amount, calculates the total operation amount, and verifies the available balance for its execution. Upon successful verification, a block is placed on the card for the full operation amount, and the available credit card balance is recalculated. Information about the successfully authorized transaction is directed to the banking system through Apache Kafka. If any verification fails, the system declines the transaction and does not perform the operation.

The system asynchronously processes messages sent by the processing system and upon receiving a message about an authorized transaction, initiates the data processing procedure. Based on transaction information (operation type, amount, account, etc.), the system determines which card agreement the operation is processed under and according to product conditions forms a new obligation within the credit limit provided for this operation, while also forming a payment schedule. If the obligation amount exceeds the available credit limit balance, the system still forms the obligation and creates a corresponding notification for the bank employee. If product conditions require parameter selection and customer confirmation for this obligation, the system forms and sends a notification to the customer in the mobile application. After receiving the message, the customer proceeds to the notification, views operation data, and selects parameters from the system-provided list (López, & Ruiz, 2020).

The settlement process completes the transaction cycle through the following algorithmic steps:

Financial Document Processing in Processing System: Financial document data is directed to the processing system through SFTP, where the received information is verified with the authorized transaction data. In case of complete match in amount and currency, the card system performs the operation of debiting funds from the account and removes the block on the transaction amount. Information about the processed financial document is directed to the banking system through Apache Kafka (Lessmann et al., 2015).

Financial Document Processing in CCMP System: The system asynchronously processes messages sent by the processing system and upon receiving a message about a financial document, initiates the data processing procedure. Based on information in the financial document, the system performs reconciliation with data of the created obligation received during authorization. In case of complete match in amount and currency, and if the obligation creation date matches the financial document date, the system performs confirmation operation (Dragičević, & Delibašić, 2021). If amount and currency match but the financial document date is later than the obligation creation date, the system updates the obligation, reformulating the payment schedule, and then performs the confirmation operation.

Security, Compliance and Monitoring Framework

The implementation of credit limit automation systems requires robust security measures and compliance with regulatory standards across multiple jurisdictions. Financial data protection implements multiple security layers:

– Encryption Protocols: All data transmissions between systems utilize TLS 1.3 encryption for data in transit and AES-256 encryption for data at rest.

– Authentication Systems: Multi-factor authentication incorporating OTP (One-Time Password) and biometric verification ensures that customer actions are confirmed and secure.

– Access Control: Role-based access control (RBAC) systems with integration to Active Directory ensure that only authorized personnel can access sensitive financial data.

Regulatory Compliance

The architectural design incorporates compliance with major financial regulations:

– GDPR Compliance: For international operations, the system implements right to be forgotten functionality and data minimization principles.

– AML Requirements: Anti-money laundering algorithms are integrated into the transaction monitoring process, with suspicious activity reports generated automatically based on predefined patterns.

– Credit Reporting Standards: Integration with credit bureaus follows standardized protocols for data exchange and reporting.

System Monitoring and Maintenance

Continuous monitoring ensures system reliability and performance:

– Performance Metrics: Response times, error rates, and system availability are tracked in real-time using Prometheus and Grafana dashboards.

– Audit Logging: All critical operations are logged to Elasticsearch for future audit and analysis, with retention policies aligned with regulatory requirements.

– Incident Response: Automated alerting systems notify operations teams of anomalies, with escalation procedures based on severity levels.

Current Implementation Challenges

Based on our analysis, the main challenges in credit limit automation include:

– Data Quality and Availability: Inconsistent data quality from external sources (AKB, Asan Finance) can lead to suboptimal credit decisions. Future systems must incorporate data validation mechanisms and alternative data sources;

– Real-time Processing Requirements: The need for immediate credit decisions conflicts with the batch-oriented nature of some risk assessment models. Stream processing architectures require further development;

– Regulatory Compliance: Evolving regulatory landscapes create challenges for maintaining compliant systems across different jurisdictions. Adaptive regulatory frameworks must be incorporated into system designs;

– System Integration Complexity: Heterogeneous legacy systems in banking environments create integration challenges that increase implementation time and costs. Standardization efforts across the industry would address this challenge.

The digital transformation of the banking sector has elevated customer expectations for personalized and flexible financial products. Central to any credit product, be it a traditional loan or a credit card limit, is the repayment schedule. This document not only legally binds the customer and the bank but also serves as the primary source of truth for cash flow forecasting, interest income recognition,

and customer communication. Traditional, hard-coded solutions for schedule generation are inherently rigid, making it costly and time-consuming to adapt to new product offerings, regulatory changes, or individualized customer terms. Consequently, there is a pressing need for a next-generation scheduling system that prioritizes configurability, flexibility, and automation. This paper outlines the architectural and functional requirements for such a system and provides a formalized algorithm for one of the most common repayment models.

A modern repayment scheduling system must be designed as a centralized, rules-based engine within the banking software ecosystem. Its primary objective is to calculate and manage the sequence of payments over the life of a credit obligation. The following non-negotiable requirements have been identified:

Parameterized Configuration without Code Changes: The system must provide a user-friendly interface (e.g., an admin portal) for business analysts and product managers to define and modify calculation parameters. This includes setting interest rates, fees, payment frequencies, and day count conventions without requiring software deployment cycles. This agility is crucial for rapid product innovation and competitive response.

Support for Heterogeneous Repayment Models: The engine must be capable of supporting a wide array of repayment methodologies to cater to different product types and customer preferences. These models must include:

Annuity/EMI (Equated Monthly Installment): Fixed periodic payments covering both principal and interest;

Installment (Straight-Line Principal): Fixed principal portions with interest calculated on the remaining balance, leading to decreasing total payments over time;

Minimum Payment (Revolving Credit): A small percentage of the outstanding balance plus fees and interest, typical for credit cards;

Fixed Monthly Amount: A predetermined fixed amount paid towards the balance, common for informal arrangements.

The system must automatically trigger a recalculation of the future payment schedule upon the occurrence of specific events. These events include contract modifications (e.g., a change in interest rate),

partial prepayments, or full early settlements. The recalculated schedule must be instantly available to all downstream systems.

A critical value-added feature is the integration with a centralized notification system (e.g., via message queues like Apache Kafka). Upon generating or updating a schedule, the engine should push events to trigger automated communications (SMS, email, in-app notifications) to inform customers of upcoming payment due dates, amounts, and any changes to their schedule.

The algorithm must correctly handle both partial and full prepayments. For a partial prepayment, the system should recalculate the amortization schedule, potentially offering the customer the choice to either reduce the monthly payment amount while keeping the term constant or reduce the loan term while keeping the installments constant.

To ensure consistency and simplify calculations across all payment types, a standard temporal basis must be adopted. The system should employ the 30/360 day count convention, which assumes that each month consists of 30 days and each year consists of 360 days. This convention eliminates the complexities associated with varying month lengths and leap years, ensuring predictable and uniform interest calculations for all customers and products that utilize it. This is mathematically expressed by basing monthly calculations on 1/12 of the annual rate. The annuity payment model is one of the most prevalent structures for installment loans. It is characterized by a constant periodic payment amount A over the term N, with the proportion of interest to principal within each payment shifting over time.

Discussion and conclusions

This research has examined the architectural frameworks and algorithmic interactions that enable automated credit limit establishment in modern banking systems. Our analysis demonstrates that effective credit limit automation requires integration of multiple heterogeneous systems through standardized APIs and message brokering systems.

The hybrid architecture approach combining microservices principles with event-driven communication patterns represents the current state-of-the-art in banking system design. This architecture

enables real-time risk assessment while maintaining system reliability and regulatory compliance. The implementation of such systems directly impa cts financial inclusion, credit portfolio quality, and operational efficiency of financial institutions.

Future research should focus on artificial intelligence integration, alternative data utilization, and blockchain applications to further enhance the effectiveness of credit limit automation systems. Additionally, industry-wide API standardization efforts would reduce integration complexity and accelerate innovation in financial services.

The advancement of automated credit limit establishment systems represents a critical component of the digital transformation of banking, with significant implications for economic growth, financial inclusion, and risk management in the financial sector.

References

Anderson, P. (2023). API economy in banking integration. *Journal of Financial Technology, 15*(4), 78–95.

Dragičević, M., & Delibašić, B. (2021). Designing a microservices architecture for a large-scale financial system: A case study. *Journal of Systems and Software, 182*, 111088. https://doi.org/10.1016/j.jss.2021.111088.

Johnson, M. (2024). Real-time risk assessment in credit decisioning. *Financial Innovation Review, 8*(2), 112–134.

Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update. *European Journal of Operational Research, 247*(1), 124–136.

López, J. C., & Ruiz, S. (2020). Real-time risk analytics in digital lending: Architectures and challenges. *Journal of Financial Transformation, 51*, 145–156.

Richardson, C., & Smith, M. (2019). API-driven banking: The foundation of digital financial innovation. *International Journal of Financial Innovation in Banking, 2*(3), 213–230.

Smith, J. (2024). Microservice architectures in financial services. *Journal of Banking Technology, 12*(3), 45–67.

Williams, K. (2023). Machine learning applications in consumer credit scoring. *Risk Management Today, 22*(1), 33–57.

References

Anderson, P. (2023). API economy in banking integration. *Journal of Financial Technology, 15*(4), 78–95.

Dragičević, M., & Delibašić, B. (2021). Designing a microservices architecture for a large-scale financial system: A case study. *Journal of Systems and Software, 182*, 111088. https://doi.org/10.1016/j.jss.2021.111088.

Johnson, M. (2024). Real-time risk assessment in credit decisioning. *Financial Innovation Review, 8*(2), 112–134.

Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update. *European Journal of Operational Research, 247*(1), 124–136.

López, J. C., & Ruiz, S. (2020). Real-time risk analytics in digital lending: Architectures and challenges. *Journal of Financial Transformation, 51*, 145–156.

Richardson, C., & Smith, M. (2019). API-driven banking: The foundation of digital financial innovation. *International Journal of Financial Innovation in Banking, 2*(3), 213–230.

Smith, J. (2024). Microservice architectures in financial services. *Journal of Banking Technology, 12*(3), 45–67.

Williams, K. (2023). Machine learning applications in consumer credit scoring. *Risk Management Today, 22*(1), 33–57.

Отримано редакцією журналу / Received: 24.09.25
Прорецензовано / Revised: 25.09.25
Схвалено до друку / Accepted: 01.10.25

Рамін САМАДОВ, канд. техн. наук
ORCID ID: 0009-0002-9384-6541
e-mail: ramin.samedov@gmail.com
Бакинський державний університет, Баку, Азербайджан

АРХІТЕКТУРА ТА АЛГОРИТМИ ВЗАЄМОДІЇ БАНКІВСЬКИХ І ПЛАТІЖНИХ СИСТЕМ У ПРОЦЕСІ АВТОМАТИЗОВАНОГО ВСТАНОВЛЕННЯ КРЕДИТНОГО ЛІМІТУ

У цій статті розглядаються архітектурні підходи та алгоритмічні взаємодії між банківськими і платіжними системами, які забезпечують автоматизоване прийняття кредитних рішень. На основі системного аналізу сучасної банківської інфраструктури визначено ключові компоненти, зокрема скорингові рушії, процесингові центри та платформи управління картками, що взаємодіють через REST API та асинхронні системи обміну повідомленнями (Apache Kafka). Дослідження показує, що сучасні системи автоматизації кредитних лімітів використовують гібридні архітектури, які поєднують принципи мікросервісів із подієво-орієнтованими моделями комунікації, забезпечуючи оцінку ризиків у реальному часі при дотриманні регуляторних вимог. Результати свідчать, що впровадження таких систем безпосередньо впливає на фінансову інклюзію, якість кредитного портфеля та операційну ефективність фінансових установ.

К л ю ч о в і с л о в а : ***кредитний скоринг, банківська архітектура, управління ризиками, фінансові технології, інтеграція API, платіжні системи, мікросервіси, обробка в режимі реального часу.***

Автор заявляє про відсутність конфлікту інтересів. Спонсори не брали участі в розробленні дослідження; у зборі, аналізі чи інтерпретації даних; у написанні рукопису; в рішенні про публікацію результатів.

The author declares no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.