

Development of an Automated Tax Deadline Notification System via LINE for Accounting Firms Using Event-Driven and Dynamic Rule-Based Architecture

Chanapat Kitpatcharakulchot
College of Arts, Media and Technology
Chiang Mai University
Chiang Mai, Thailand
k.chanapat@gmail.com

Kittitouch Suteeca, Ph.D.
College of Arts, Media and Technology
Chiang Mai University
Chiang Mai, Thailand
kittitouch.s@cmu.ac.th

Abstract—This study presents the Automated Tax Deadline Notification System via LINE for Accounting Firms Using Event-Driven and Dynamic Rule-Based Architecture, a web-based system that calculates tax filing deadlines from configurable rules and notifies staff and supervisors through LINE when task-state events occur. Exploratory interviews with a supervisor and two accounting staff at an accounting firm in Chiang Mai, Thailand identified problems that recur in every monthly and annual filing cycle: misremembered deadlines for clients with non-standard fiscal years, spreadsheet formula errors and public holidays ignored in manual date calculation, and no central view of filing status. Tax rules are stored as data, so supervisors can add or modify rules without changing source code or redeploying the system. A daily scheduled event triggers rule evaluation and task generation, and an alert is sent only when a task meets a state condition, such as being unfiled seven days or one day before its due date; a task still unfiled on or after its due date is escalated to the supervisor. Development followed ISO/IEC TR 29110-5-1-2:2011 over 24 weeks. All 134 functional and non-functional test cases passed, including tests confirming that edited rules take effect in the next calculation without a restart, and User Acceptance Testing (UAT) with three real users across 21 scenarios met the 100% pass criterion. The supervisor's time to configure tax rules fell from 300 s to 120 s over three trials. Because firm-specific knowledge is held as data, the design can be reused by other firms and for other deadline-driven compliance tasks.

Keywords—accounting firm, dynamic rule-based engine, event-driven architecture, ISO/IEC 29110, LINE Messaging API, tax deadline notification

I. INTRODUCTION

In professional accounting services, temporal accuracy is a key performance indicator that determines the credibility of a firm. A recurring difficulty in practice is the management of non-standard fiscal years, that is, reporting periods that do not align with the calendar year or that close on client-specific dates such as 15 February or 20 January. Each client may also be subject to different tax forms, filing methods (paper or e-Filing), and deadline rules, which makes manual tracking error-prone.

To understand the problem, exploratory interviews were conducted at an accounting firm in Chiang Mai, Thailand, with the supervisor who oversees filing work and two accounting staff who prepare the filings. The firm tracked deadlines in Microsoft Excel and on paper. The supervisor reported that staff misremembered due dates because clients had different fiscal years, that Excel formulas shifted and produced wrong dates that had to be rechecked by hand, that public holidays were not considered when due dates were calculated, which risked filing

on the wrong date and incurring penalties, and that the status of each staff member's work could not be followed in one place. The staff asked for automatic due-date calculation that accounts for holidays, a clear order of urgency, and alerts on LINE, which they already use every day.

These problems are not occasional; they follow from the structure of the workload. Each corporate client generates several filings every month, such as withholding tax (P.N.D. 1, 3, and 53), value-added tax (P.P. 30), and social security contributions, each due on a fixed day of the following month, and further annual obligations, such as corporate income tax (P.N.D. 50), the shareholders' meeting, and financial statements, whose deadlines are counted from that client's own fiscal year end. A firm with N clients therefore tracks several times N deadlines every month, and the annual deadlines fall on different dates for each client. Every new client adds to this load, which is why the firm described the problems as recurring every year and growing with its client base.

A comparative review of five task-management tools commonly used in Thailand, namely Monday.com, Trello, ClickUp, Google Calendar, and Microsoft Excel, was conducted to identify gaps (Table I). All of them support basic task tracking, and most provide reminders and team collaboration. However, none provides a built-in tax rule base or calculates statutory due dates, and none raises alerts based on the state of a tax task.

TABLE I
COMPARISON OF EXISTING TOOLS AND THE PROPOSED SYSTEM

Feature	Mon.	Trel.	Click	GCal	Excel	Proposed
Task tracking	✓	✓	✓	✓	✓	✓
Reminder / alert	✓	✓	✓	✓	X	✓
Team collaboration	✓	✓	✓	✓	Partial	✓
Recurring tasks	✓	Partial	✓	✓	X	✓
Statutory due-date rules	X	X	X	X	X	✓
Alerts based on tax conditions	X	X	X	X	X	✓

To address these gaps, this study developed the Automated Tax Deadline Notification System via LINE for Accounting Firms Using Event-Driven and Dynamic Rule-Based Architecture (hereafter, the proposed system). The objectives were to: (1) build a central database that supports flexible fiscal years and stores filing evidence; (2) develop a deadline calculation engine based on dynamic rules; (3) deliver multi-stage notifications to staff and supervisors via the LINE Messaging API; (4) apply an Event-Driven Architecture (EDA)

to detect task status and dispatch notifications automatically every day at 08:00; and (5) provide a real-time operational dashboard, all developed in accordance with ISO/IEC 29110.

II. RELATED WORKS

A. Concepts and Theories

Event-Driven Architecture. EDA is an architectural style in which events, rather than a fixed sequence of calls, drive processing. It comprises event producers, an event channel, and event consumers. Because modules react only to the events relevant to them, the system is loosely coupled and individual modules can be extended or replaced without affecting the overall structure [1].

Dynamic Rule-Based Engine. A rule-based engine separates business rules from application logic. In a static approach, rules are hard-coded and every change requires recompilation and redeployment. In a dynamic approach, rules are stored as data and evaluated at runtime, so business users can add or change rules immediately [2].

Modified Due-Date (MDD) Rule. Baker and Bertrand [3] proposed the MDD rule, a dynamic priority rule that ranks jobs by comparing each job's due date with its earliest possible completion time, and which adapts better than static rules under heavy workloads.

LINE Messaging API and ISO/IEC 29110. The LINE Messaging API pushes automated messages, including Flex Messages with buttons, and LIFF opens web pages inside LINE [4]. ISO/IEC 29110 is a software engineering standard for Very Small Entities whose Basic Profile defines Project Management (PM) and Software Implementation (SI) processes [5].

B. Relevant Studies

Chockalingam [1] described the shift of accounting from batch processing to event-driven, real-time processing and noted that moving from manual to digital accounting can improve data processing accuracy by 65–85%. Yue *et al.* [2] proposed a dynamic rule engine for real-time risk control in financial services that allows rules to be updated without recompiling the system. Together, these studies support combining EDA with a dynamic rule engine for deadline-driven accounting work.

III. SYSTEM DESIGN AND ARCHITECTURE

A. System Design

The proposed system supports two user roles, accounting staff (Staff) and supervisors (Supervisor), plus automated functions. Requirements were documented as 33 use case descriptions and refined into 65 functional requirements (FR-01 to FR-65) in 12 modules and 8 non-functional requirements (NFR-01 to NFR-08). Supervisors manage clients and fiscal years, tax rules, holidays, users, and task assignments, and view a team dashboard; staff view a personal dashboard, update task status from pending to in progress to filed, upload filing evidence, and link a LINE account.

B. Event-Driven Notification

The proposed system treats a notification as the consequence of an event in the state of a tax task, not as a reminder set for a fixed date. Events are of two types: time-based events, published by a scheduled cron job as a daily-check event at 08:00, and user-triggered events, such as a supervisor

reassigning a task or pressing a follow-up button, or a message such as an account-linking code received through the LINE Webhook. On the daily-check event (Fig. 1), the Rule Engine loads rules from the Rule Base, recalculates due dates with holiday shifting, and re-ranks task urgency; the Task Generator creates or updates tasks; and the Notification Module evaluates each task against the conditions in Table II and sends a LINE message only when a condition is met.

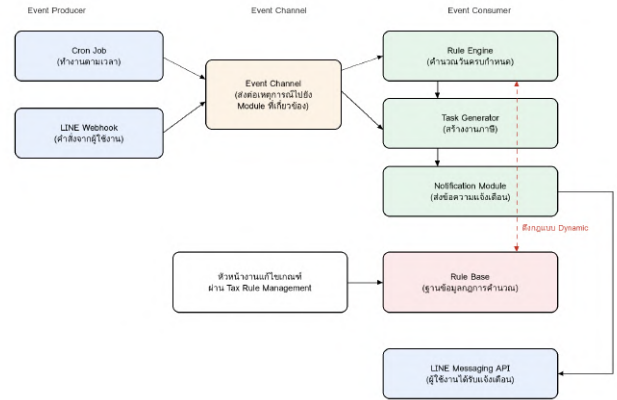


Fig. 1. Event flow integrating EDA with the Dynamic Rule-Based Engine.

TABLE II
NOTIFICATION EVENTS, CONDITIONS, AND RECIPIENTS

Event	Condition	Recipient
Daily summary	Staff has pending tasks (skipped if none)	Staff
Advance reminder (7 days)	Task not filed; due in 7 days	Assigned staff
Advance reminder (1 day)	Task not filed; due in 1 day	Assigned staff and supervisor
Escalation	Task not filed on or after its due date	Supervisor
Task reassigned	Assignee changed by supervisor	New assignee
Follow-up	Supervisor presses “follow up”	Assigned staff
Delivery failure	Three retries at 5-min intervals fail	Supervisor
Account linking	Linking code received via Webhook	Sender

Because each condition is evaluated against the current task state and the recalculated due date, a filed task generates no further alerts, a task still open at its due date is escalated to a second person, and a rule or holiday change is reflected in the next day's alerts without rescheduling. A calendar or e-mail reminder, by contrast, fires on the date it was set regardless of whether the work is done and does not escalate. The contribution therefore lies in when and to whom a message is sent, not only in the channel. LINE was chosen because staff already use it daily, and each Flex Message links directly to the task through LIFF. Communication is currently one-way, a limitation discussed in Section VI.

C. Dynamic Rule-Based Engine

Each rule is stored as a database record that specifies the tax form, a reference date, a calculation method, a value, and the legal provision on which it is based (Table III, Fig. 2). Three calculation methods are supported (a fixed day of the following month, an offset in days, and an offset in months) together with three reference dates (the end of the tax month, the client's fiscal year end, and the date of the shareholders' meeting). Monthly filings use a fixed day, while annual obligations use

offsets from each client’s fiscal year end, so non-standard fiscal years need no special handling. A calculated date that falls on a weekend or public holiday is moved to the next working day. When a supervisor edits a rule, the next calculation uses the new value; the system warns if existing tasks depend on the rule, and those tasks keep their dates. Task urgency is ranked using an MDD-based priority score.

TABLE III
EXAMPLES OF RULES IN THE RULE BASE

Rule	Obligation	Method	Value	Reference
R-01	P.N.D. 1	Fixed day	15th	End of tax month
R-04	P.P. 30 (VAT)	Fixed day	23rd	End of tax month
R-07	Shareholders’ meeting	Offset months	+4	Fiscal year end
R-09	P.N.D. 50	Offset days	+150	Fiscal year end
R-10	Shareholder list (BOJ. 5)	Offset days	+14	Meeting date

เพิ่มเกณฑ์ใหม่

รหัสเกณฑ์ *

แบบฟอร์มภาษี

R-16

ก.จ.ค.50

ชื่อรายการ *

ภาษีเงินได้บุคคล

วิธีคำนวณ *

+N Days

จำนวนวัน *

150

เริ่มนับจาก *

สิ้นรอบบัญชี

คำอธิบาย (ไม่บังคับ)

ยื่นภายใน 150 วันหลังสิ้นรอบบัญชี

กฎหมายอ้างอิง *

ป.ร.ศ.ฎการ ม.67 พ.ร.

ยกเลิก

เพิ่มเกณฑ์

Fig. 2. Adding a new tax rule on the Tax Rule Management page.

D. System Architecture

The system follows a four-layer architecture (Fig. 3) built on the Next.js 14 App Router: a presentation layer, an API layer with role-based access control and Zod validation, a business logic layer (rule engine, task generator, holiday-safe date shifting, and notification service), and a data access layer using the Repository Pattern with Prisma ORM over PostgreSQL 16. Rules are stored using PostgreSQL JSONB, and a separate database stores system, error, and audit logs. The Google Calendar API is used only to synchronize Thai public holidays. Scheduled jobs run in an Alpine Linux container and call protected /api/cron endpoints at 08:00 (UTC+7); all timestamps are stored in UTC.

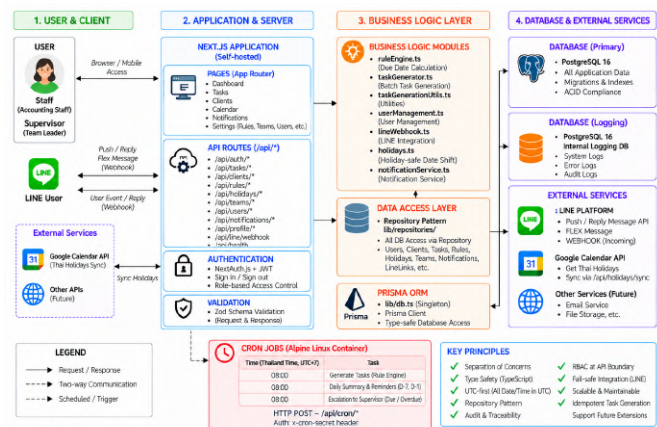


Fig. 3. System architecture of the proposed system.

IV. IMPLEMENTATION

Development followed ISO/IEC TR 29110-5-1-2:2011 over 24 weeks (February to July 2026), covering the PM and SI processes. Work products included the Project Plan, SRS, SDD, Conflict Management Plan, Test Plan, Test Record, Traceability Record, and User Manual. The frontend uses Next.js Server Components with shadcn/ui and Recharts; authentication uses NextAuth.js v4 with JWT and bcrypt-hashed passwords. Users link their LINE account by sending a code from the profile page to the LINE Official Account, and webhook requests are verified with an HMAC-SHA256 signature. Notifications are sent as Flex Messages (Fig. 4) and recorded in a notification history; users without a linked LINE account are skipped and logged.



Fig. 4. Tax deadline notifications delivered as LINE Flex Messages.

V. CASE STUDY AND EVALUATION

A. System Testing

Test cases were derived from the FR and NFR identifiers in the SRS, so that every requirement had at least one test, and were mapped through the Traceability Record. Unit testing used Postman for every API endpoint, and system testing was black-box, scripted from the use cases. All 115 functional test cases across 12 modules, including 11 notification cases covering each event in Table II, and all 19 non-functional test cases passed after three minor defects in task filtering and evidence upload were fixed. A traceability review confirmed that all 65 FRs and 8 NFRs were implemented and tested.

B. Verification of Rule Configuration

Because changing rules without changing code is the main claim of the dynamic rule engine, it was verified at the unit, system, and user levels (Table IV).

TABLE IV
VERIFICATION OF RULE CONFIGURATION WITHOUT CODE CHANGES

Test ID	What was verified	Result
ENG-02-04	Fixed day, offset days, and offset months give correct dates	Pass
ENG-05-07	Weekend, holiday, and three consecutive holidays shift to next working day	Pass
ENG-08, NFR-07-01	Edited value used in next calculation without redeploy or restart	Pass
RULE-02, 06	Supervisor adds a rule (usable at once) and deletes a rule	Pass
RULE-04	Editing a rule linked to existing tasks shows a warning first	Pass
RULE-07, 08	All three methods and three reference dates can be saved	Pass
UAT-S05	Supervisor adds, edits, and deletes rules in real use	Pass

In UAT-S05, the supervisor initially had little understanding of how rules are defined but, after a short explanation, configured them correctly; completion time fell from 300 s to 240 s and 120 s over three trials. In practice, when the Revenue Department changes a filing deadline or the firm takes on a client with a new obligation, the supervisor updates one rule record and every subsequent task follows it, without waiting for a developer or a new release.

C. User Acceptance Testing

UAT was conducted with three real users, one supervisor and two accounting staff, across 21 scenarios (12 for the supervisor and 9 for staff) with a pass criterion of 100%. The first round produced two change requests: filtering tasks by due date rather than by task period and raising the evidence limit from one to five files per task; after the changes, all scenarios passed. Post-test interviews indicated high satisfaction: users found the system easy to learn without training and reported that it reduced the burden of remembering and tracking deadlines compared with Excel or handwritten notes.

VI. RESULT AND DISCUSSION

The proposed system met all five objectives. Deadlines for standard and non-standard fiscal years were calculated from rules that the supervisor maintained without developer involvement, and alerts were driven by task state, with escalation when work remained open. The supervisor suggested two refinements: a status for months in which a client has no filing for a given tax type, and a configurable advance-reminder period instead of a fixed seven days.

Applicability to other firms and cases. The design separates what is specific to the case firm from what is general. Firm-specific knowledge (tax forms, deadline rules, client fiscal years, and holidays) is held as data, while the event flow and rule engine are generic. Another accounting firm can therefore adopt the system by deploying the containerized application, creating its own LINE Official Account, entering its clients and fiscal year ends, and reusing or adjusting the Thai tax rules, without changing code. The same pattern, in which a due date is derived from an anchor date by a rule and followed by state-based reminders and escalation, also applies to other deadline-

driven compliance work, such as license and permit renewals, social security filings, contract renewals, and audit schedules, where only the rule base and reference dates change. Outside Thailand, the rule base and holiday calendar would be replaced, and because notification is decoupled through events, LINE could be replaced by another messaging channel.

Limitations. The evaluation involved three users at a single firm and did not measure filing timeliness or error rates before and after deployment. The comparison in Table I covered general task-management tools only; the system was not compared with accounting or ERP software that includes tax compliance modules for larger organizations, which may already calculate statutory due dates. Finally, LINE communication is one-way, so users must open the web application to update a task.

VII. CONCLUSION AND FUTURE WORK

This paper presented the Automated Tax Deadline Notification System via LINE for Accounting Firms Using Event-Driven and Dynamic Rule-Based Architecture, which calculates filing deadlines from rules that supervisors change without modifying code and sends LINE alerts when task-state events occur. Developed under ISO/IEC 29110, the system passed all 134 test cases and 21 UAT scenarios. Future work includes two-way interaction in LINE, such as marking a task as filed or acknowledging an alert from the message; a “no filing required” status; configurable reminder periods; a comparison with tax compliance software used by larger organizations; and evaluation at additional firms to measure the effect on filing timeliness.

ACKNOWLEDGMENT

The authors thank the supervisor and accounting staff of the case-study accounting firm in Chiang Mai for their participation in the interviews and user acceptance testing.

REFERENCES

- [1] D. Chockalingam, “Event-driven accounting transformation: From batch processing to real-time intelligence,” *International Journal of Computational and Experimental Science and Engineering*, vol. 11, no. 4, pp. 7491–7502, 2025, doi: 10.22399/ijcesen.4052.
- [2] L. Yue, Z. Su, and X. Dong, “Dynamic rule engine for real-time risk control in financial digital transformation,” *IEEE Xplore*, doc. 11268029, 2025. [Online]. Available: <https://ieeexplore.ieee.org/document/11268029>
- [3] K. R. Baker and J. W. M. Bertrand, “A dynamic priority rule for scheduling against due-dates,” *Journal of Operations Management*, vol. 3, no. 1, pp. 37–42, 1982.
- [4] LY Corporation, “Messaging API overview,” LINE Developers. [Online]. Available: <https://developers.line.biz/en/docs/messaging-api/overview/>
- [5] *Software Engineering — Lifecycle Profiles for Very Small Entities (VSEs) — Part 5-1-2: Management and Engineering Guide: Generic Profile Group: Basic Profile*, ISO/IEC TR 29110-5-1-2:2011.