

BAB 2

TINJAUAN PUSTAKA

2.1. Studi Sebelumnya

Penelitian sebelumnya yang dilakukan oleh Puspaningrum di Universitas Muhammadiyah Gresik dengan tujuan untuk mengintegrasikan dua sistem utama, yaitu SIAKAD dan SPADA, yang sebelumnya berjalan secara terpisah [7]. Ketidakterhubungan kedua sistem menyebabkan duplikasi data serta inefisien proses akademik. Metode yang digunakan adalah *Object Oriented Analysis and Design* (OOAD) dengan pendekatan *bottom-up*, serta pemodelan sistem dengan menggunakan *Unified Modeling Language* (UML) seperti *use case*, *activity*, dan [7]*sequence diagram*. Hasil penelitian ini menunjukkan bahwa penerapan OOAD dapat mempermudah proses integrasi tanpa harus merombak struktur sistem yang sudah berjalan. Integrasi yang dirancang tanpa harus merombak stuktur sistem yang sudah berjalan. Integrasi yang dirancang mampu meningkatkan efisiensi pengelolaan data akademik, menurunkan kesalahan input data, dan memudahkan dosen serta mahasiswa dalam mengakses layanan akademik melalui satu portal terintegrasi [7].

Selanjutnya penelitian [8], berfokus pada pengembangan sistem pendataan dan pengelolaan data penduduk di Desa Tanjung Kuyo [8]. Penelitian ini bertujuan untuk menciptakan sistem berbasis *web* yang efisien dan akurat dalam mengelola data kependudukan, menggantikan sistem manual yang sering menyebabkan duplikasi data dan keterlambatan pembaruan informasi. Metode yang digunakan adalah OOAD dengan pendekaan *waterfall* (SDLC), meliputi tahapan identifikasi masalah, analisis kebutuhan, perancangan, implementasi, pengujian, serta pemeliharaan. Penelitian ini menghasilkan Kesimpulan bahwa sistem mampu menyediakan pendataan, pencarian cepat, pembaruan informasi, serta pelaporan otomatis. Pengujian *Black Box* menunjukkan semua fitur berjalan sesuai kebutuhan pengguna. Kombinasi OOAD dan SDLC terbukti memberikan kerangka sistematis sekaligus desain yang fleksibel dan mudah dikembangkan [8].

Selanjutnya penelitian [9], dalam merancang sistem informasi digital di lingkungan pemerintahan desa, khususnya untuk pengelolaan administrasi dan publikasi informasi masyarakat [9]. Metode OOAD digunakan untuk memodelkan sistem berbasis objek, disertai pemanfaatan UML (*Use Case*, *Class*, dan *Sequence Diagram*). Hasil penelitian ini mengemukakan bahwa sistem yang dikembangkan dapat meningkatkan transparansi pelayanan publik, mempercepat proses administrasi seperti pembuatan surat keterangan, serta memungkinkan warga untuk memantau status layanan secara daring. Pendekatan OOAD terbukti efektif dalam menciptakan desain sistem yang fleksibel, *modular*, dan mudah dikembangkan [9].

Kemudian penelitian yang dilakukan oleh Riyadi, yang membahas tentang permasalahan pengelolaan penjualan konvensional pada percetakan UPrint di Surabaya [10]. Tujuannya adalah merancang sistem *e-commerce* berbasis *web* untuk meningkatkan efisiensi transaksi dan memperluas jangkauan pemasaran. Metode OOAD digunakan untuk meningkatkan efisiensi proses penjualan, mengurangi kesalahan pencatatan, dan memberikan kemudahan bagi pelanggan dalam melakukan transaksi secara daring. OOAD terbukti mampu menghasilkan rancangan sistem yang *modular* dan mudah dimengerti oleh pengembang maupun pengguna [10].

Lalu penelitian [11], membahas integrasi modul portofolio pada aplikasi MyITS *Human Capital Management* dengan sistem SISTER ITS menggunakan arsitektur *event-driven* berbasis Apache Kafka. Hasil penelitian menunjukkan bahwa pendekatan *event-driven* mampu mengintegrasikan data portofolio antar sistem secara asinkron dan meningkatkan efisiensi sinkronisasi data, meskipun ditemukan kendala pada ketergantungan terhadap stabilitas *endpoint* API sistem eksternal. Keterkaitan penelitian tersebut dengan penelitian ini terletak pada upaya integrasi antar sistem yang telah berjalan. Namun pendekatan *event-driven* memiliki kompleksitas dalam pelacakan alur proses bisnis dan audit data, sehingga kurang sesuai untuk sistem perbankan yang membutuhkan konsistensi dan kontrol ketat.

Penelitian lain [12] yang mengimplementasikan arsitektur *microservices* dalam pengembangan sistem informasi karyawan berbasis web di lingkungan BMKG. Hasil penelitian ini menunjukkan bahwa penerapan *microservices* mampu meningkatkan skalabilitas, fleksibilitas, dan kemudahan pemeliharaan sistem melalui pemisahan layanan absensi dan pengajuan cuti ke dalam *service* yang independen. Meskipun demikian, arsitektur *microservices* memerlukan kompleksitas implementasi dan pengelolaan infrastruktur yang lebih tinggi. Oleh karena itu, pendekatan ini dinilai kurang sesuai untuk penelitian ini yang berfokus pada integrasi sistem eksisting tanpa perubahan arsitektur secara menyeluruh, sehingga lebih tepat menggunakan metode OOAD untuk menjaga struktur, dokumentasi, dan keterlacakan proses bisnis.

Serta sebuah penelitian [13] di PT. Bank Bukopin, bertujuan untuk menganalisis dan merancang sistem informasi layanan pengaduan kartu kredit agar proses penanganan keluhan nasabah menjadi lebih efisien dan terkomputerisasi [13]. Penelitian ini menggunakan metode *Object Oriented Analysis and Design* (OOAD) dengan alat bantu *Unified Modeling Language* (UML). Melalui pendekatan ini, penulis memodelkan *flowmap* sistem berjalan dan sistem usulan untuk menggambarkan proses penanganan pengaduan dari awal hingga selesai. Hasil yang dicapai dari penelitian ini adalah, OOAD mampu menghasilkan rancangan sistem yang terstruktur dan *modular*, meminimalkan proses manual, serta meningkatkan

efisiensi penanganan pengaduan. Keterkaitan dengan penelitian ini terletak pada konteks dan metode yang sama. Dengan metode dan objek yang sama, penelitian Syah memperkuat alasan pemilihan metode OOAD karena terbukti efektif dalam integrasi dan otomasi proses bisnis di lingkungan perbankan [13].



Table 2.1 Perbandingan Studi Sebelumnya

No	Penulis	Tahun	Metode	Hasil
1	Puspaningrum, dkk [7].	2025	<i>Object Oriented Analysis and Design</i>	Penerapan metode OOAD mempermudah proses integrasi tanpa harus merombak struktur sistem yang sudah berjalan, serta mampu meningkatkan efisiensi pengelolaan data akademik, menurunkan kesalahan input data, dan mempermudah dosen dan mahasiswa dalam mengakses layanan akademik melalui satu portal terintegrasi.
2	Walhidayat, dkk [8].	2025	<i>Object Oriented Analysis and Design, Software Development Life Cycle</i>	Kombinasi metode OOAD dan SDLC terbukti memberikan kerangka sistematis sekaligus desain yang fleksibel dan mudah dikembangkan. Melalui metode tersebut, sistem mampu menyediakan pendataan, pencarian cepat, pembaruan informasi, serta pelaporan otomatis.
3	Mahfuzhi, dkk [9].	2025	<i>Object Oriented Analysis and Design</i>	Pendekatan OOAD terbukti efektif dalam menciptakan desain sistem yang fleksibel, <i>modular</i> , dan mudah dikembangkan. Sistem yang diciptakan dapat meningkatkan transparansi pelayanan public, mempercepat proses administrasi seperti pembuatan surat keterangan, serta memungkinkan warga untuk memantau status layanan secara daring.
4	Riyadi, dkk [10].	2024	<i>Object Oriented Analysis and Design</i>	Hasil penelitian ini mengemukakan bahwa sistem yang dikembangkan dapat meningkatkan transparansi pelayanan publik, mempercepat proses administrasi seperti pembuatan surat keterangan, serta memungkinkan warga untuk memantau status layanan secara daring. Pendekatan OOAD terbukti efektif dalam menciptakan desain sistem yang fleksibel, <i>modular</i> , dan mudah dikembangkan
5	Adila, dkk [11].	2022	Event Driven	Hasil penelitian menunjukan bahwa pendekatan <i>event-driven</i> mampu mengintegrasikan data portofolio antar sistem secara asinkron dan

No	Penulis	Tahun	Metode	Hasil
				meningkatkan efisiensi sinkronisasi data, meskipun ditemukan kendala pada ketergantungan terhadap stabilitas <i>endpoint</i> API sistem eksternal.
6	Hanifah, dkk [12].	2025	Architecture Microservices	Hasil penelitian ini menunjukkan bahwa penerapan <i>microservices</i> mampu meningkatkan skalabilitas, fleksibilitas, dan kemudahan pemeliharaan sistem melalui pemisahan layanan absensi dan pengajuan cuti ke dalam <i>service</i> yang independen. Meskipun demikian, arsitektur <i>microservies</i> memerlukan kompleksitas implementasi dan pengelolaan infrastruktur yang lebih tinggi.
7	Syah, dkk [13].	2022	<i>Object Oriented Analysis and Design</i>	Hasil penelitian ini menunjukkan bahwa penerapan metode OOAD mampu menghasilkan rancangan sistem yang terstruktur dan <i>modular</i> , meminimalkan proses manual, serta meningkatkan efisiensi penanganan pengaduan

2.2. Dasar Teori

2.2.1. Web E-Ticketing PT. BOM

PT Bank Open Money adalah salah satu bank terkemuka di Indonesia yang memiliki komitmen tinggi terhadap penyimpanan uang masyarakat Indonesia. PT Bank Open Money telah menjadi kepercayaan masyarakat Indonesia sejak lebih dari 25 tahun. Saat ini, PT BOM memiliki lima grup dalam sektor Teknologi Informasi (TI), dengan lebih dari 10 subgrup di dalamnya, yang secara keseluruhan memiliki lebih dari 24.000 karyawan (2025).

PT BOM dalam sektor IT, menggunakan sistem *web* bernama E-Ticketing untuk mengelola berbagai aktivitas internal di IT PT BOM, secara khusus untuk mengelola *request* kebutuhan *tangible*, seperti server, CCTV, *data center*, dan juga *intangible* seperti pembelian *web server*, peningkatan kapasitas basis data, dll. *Web* E-Ticketing ini juga menjadi salah satu fondasi atau sumber data dari *Web Back Office*

2.2.2. Web Back Office PT. BOM

Web Back Office PT. Bank Open Money (PT. BOM) merupakan sistem internal yang dikelola oleh Biro IT Architecture sejak Oktober 2024 untuk mengelola seluruh permintaan pengadaan yang masuk melalui *Web* E-Ticketing. Sistem ini berfungsi sebagai sarana dokumentasi, evaluasi, dan tindak lanjut setiap *request* pengadaan, mulai dari pembahasan kebutuhan hingga pengambilan keputusan. Namun, proses bisnis yang berjalan masih bersifat manual, di mana data dari E-Ticketing harus diinput ke *spreadsheet* dan dimasukkan ulang ke *Web Back Office*. Kondisi ini menimbulkan duplikasi data, potensi kesalahan input, serta inefisiensi waktu kerja staf biro.

Untuk mengatasi hal tersebut, dilakukan langkah integrasi antara *Web* E-Ticketing dan *Web Back Office* melalui penerapan *Application Programming Interface* (API). Dengan API, pertukaran data antar sistem dapat dilakukan secara otomatis dan *real-time* tanpa proses input ulang. Integrasi ini meningkatkan konsistensi data, mempercepat alur kerja pengadaan, serta memastikan perhitungan *Service Level Agreement* (SLA) yang lebih akurat dan seragam di kedua sistem. Selain efisiensi, penggunaan API juga membuka peluang pengembangan sistem yang lebih *modular* dan adaptif terhadap kebutuhan bisnis di masa mendatang.

2.2.3. Integrasi Sistem

Integrasi sistem adalah proses melakukan penyatuan antara dua atau lebih sistem informasi yang berbeda agar dapat bekerja secara terpadu dalam mencapai tujuan organisasi. Menurut [14], integrasi sistem merupakan langkah strategis untuk menghubungkan berbagai aplikasi bisnis agar mampu berbagi data, fungsi, dan proses secara efisien dalam satu ekosistem teknologi yang terkoordinasi [14]. Integrasi bertujuan untuk mengurangi redundansi dan inkonsistensi data, sehingga dapat mempercepat proses bisnis dan meningkatkan akurasi dan konsistensi informasi antar biro atau departemen [14].

Referensi [14] menjelaskan bahwa evolusi integrasi sistem telah melalui beberapa pendekatan, yaitu *Enterprise Application Integration* (EAI), *Business-to-Business Integration* (B2B), *Business Process Management* (BPM), dan *Service-Oriented Architecture* (SOA).

1. EAI, berfokus pada integrasi internal antara aplikasi dalam satu organisasi melalui protokol middleware yang mengelola pertukaran data dan proses bisnis.
2. B2B *Integration*, berfungsi untuk memperluas integrasi ke antar organisasi, memungkinkan pertukaran data secara aman dan otomatis antara sistem Perusahaan berbeda.
3. BPM, berfungsi menekankan sinkronisasi alur proses bisnis lintas sistem agar efisiensi organisasi meningkat.
4. OOAD, merupakan pendekatan modern yang menggunakan layanan sebagai komponen independent yang dapat diakses melalui antarmuka standar seperti API.

Pendekatan OOAD dan API menjadi dasar dari integrasi modern karena bersifat *loosely coupled*, yakni setiap sistem dapat berinteraksi tanpa perlu saling tergantung secara langsung pada struktur internal masing-masing [14].

Table 2.2 Perbandingan Metode OOAD dengan Metode Lain

Parameter	Object Oriented Analysis and Design	Microservice Architecture	Event Driven
Struktur desain sistem	Berorientasi objek (<i>class</i> , <i>package</i> , <i>layer</i>), terstruktur, dan terdokumentasi	Berorientasi layanan, terdistribusi	Berorientasi event, tidak berurutan secara linear
Pendekatan terhadap modularisasi	Modularisasi dilakukan pada level objek dan modul sistem	Modularisasi dilakukan pada level <i>service</i> independen (pembuatan aplikasi kecil)	Modularisasi dilakukan melalui <i>event producer</i> dan <i>consumer</i>
Kesesuaian untuk sistem perbankan	Sangat sesuai, karena terstruktur, terdokumentasi, dan mudah diaudit	Sesuai untuk sistem berskala besar dan terdistribusi	Kurang sesuai untuk sistem inti yang membutuhkan konsistensi dan audit ketat
Security dan Version Control	Memungkinkan kontrol ketat atas objek dan komunikasi antar sistem melalui API. Sehingga setiap API didesain memiliki batasan akses, validasi data, dan dokumentasi versi yang jelas, sehingga memudahkan pengelolaan perubahan.	Keamanan tersebar di setiap <i>service</i> , sehingga pengelolaannya lebih kompleks	Keamanan event lebih kompleks dan sulit dilacak
Komunikasi antar komponen	Sinkron melalui pemanggilan API dan interaksi objek	Sinkron / asinkron melalui REST API atau gRPC	Asinkron melalui message broker atau event queue
Scalability dan Maintenance	Tinggi, mudah dikembangkan karena sistem bersifat <i>modular</i> dan <i>reusable</i>	Sangat tinggi, namun biaya pemeliharaan besar	Tinggi, tetapi sulit dalam <i>debugging</i> dan <i>tracing</i> ketika terjadi <i>error</i>
Kompleksitas Implementasi	Sedang, fokus pada kejelasan desain dan alur sistem	Tinggi, memerlukan <i>service orchestration</i> dan <i>monitoring</i>	Tinggi, memerlukan <i>event broker</i> dan <i>tracing event</i>

Berdasarkan Tabel 2.2, metode Object-Oriented Analysis dan Design (OOAD) dibandingkan dengan arsitektur microservice dan event driven karena ketingganya memiliki fokus pada modularisasi sistem. OOAD menekankan pemodelan objek, relasi, dan perilaku sistem secara

terstruktur melalui artefak UML, sehingga sangat sesuai untuk sistem integrasi yang membutuhkan dokumentasi formal dan keterlacakan proses bisnis.

Arsitektur microservices dan event-driven menawarkan fleksibilitas dan skalabilitas tinggi, namun memiliki kompleksitas implementasi dan pengelolaan yang lebih besar. Dalam konteks penelitian ini, integrasi dilakukan pada sistem yang telah berjalan tanpa melakukan perubahan arsitektur secara menyeluruh. Oleh karena itu, pendekatan OOAD dipilih karena mampu mendukung integrasi berbasis API secara terstruktur, modular, dan sesuai dengan kebutuhan sistem perbankan.



Metode OOAD jika dibandingkan dengan metode lain seperti *Rapid Application Development* (RAD) dan *Extreme Programming* (XP), metode OOAD memiliki keunggulan untuk proyek integrasi sistem dalam lingkup departemen karena memberikan keseimbangan antara ketelitian desain dan skalabilitas jangka panjang. Alasan tersebut menjadi dasar pemilihan metode OOAD sebagai metode utama dalam penelitian ini.

2.2.4. Object Oriented Analysis and Design (OOAD)

Metode *Object Oriented Analysis and Design* (OOAD) ini merupakan metode analisis yang dikembangkan oleh Booch, seperti pada [15]. Metode OOAD merupakan penggabungan antara tiga pendekatan, yaitu Object Analysis (OOA), Object Oriented Design (OOD), dan *Object Oriented Programming* [15].

Menurut penelitian [15], OOAD menggabungkan tiga pendekatan berorientasi objek, yaitu:

1. Object-Oriented Analysis (OOA)

OOA berfokus pada pemahaman kebutuhan sistem dari perspektif konsep, kelas, dan objek dalam domain masalah. Analisis dilakukan untuk mengidentifikasi elemen-elemen utama, perilaku, hubungan antar objek, dan batasan sistem.

2. Object-Oriented Design (OOD)

OOD mengubah hasil analisis ke dalam bentuk rancangan teknis. Pada tahap ini dilakukan dekomposisi berorientasi objek, penentuan struktur sistem (arsitektur), interaksi antar objek, serta penyusunan model statis dan dinamis menggunakan notasi UML.

3. Object-Oriented Programming (OOP)

Tahap implementasi yang menerjemahkan rancangan OOD ke dalam kode program. Objek menjadi unit konstruksi utama, dan kelas menjadi template pembentuk objek, dengan hubungan hierarkis melalui *inheritance* serta interaksi melalui *messages*.

Booch dkk (2007) menjelaskan bahwa proses OOAD menghasilkan hubungan berlapis, di mana keluaran *Object-Oriented Analysis* menjadi dasar bagi *Object-Oriented Design*, dan hasil rancangan OOD berfungsi sebagai *blueprint* untuk implementasi OOP [15].

Metode OOAD menurut Booch, memiliki proses makro dan mikro, yang diimplementasikan melalui empat aktivitas utama, yaitu *Inception*, *Elaboration*, *Construction*, dan *Transition* [15].

1. Inception

Tahap *Inception* merupakan fase awal yang berfungsi untuk menetapkan landasan konseptual dari sistem yang akan dikembangkan. Fokus utama pada tahap ini adalah memahami dan merumuskan *system requirements* secara menyeluruh tanpa mempertimbangkan aspek teknis implementasi seperti bahasa pemrograman, platform, maupun *database*.

Pada tahap ini dilakukan identifikasi elemen-elemen utama, subsistem, serta fungsi-fungsi yang diperlukan oleh sistem. Tujuannya adalah memastikan bahwa ruang lingkup (*scope*) dan sasaran sistem telah didefinisikan dengan jelas sebelum masuk ke tahapan teknis berikutnya [15].

2. Elaboration

Tahap *Elaboration* merupakan fase perincian dan penajaman kebutuhan yang telah dirumuskan pada fase *Inception*. Pada tahap ini dilakukan pendalaman arsitektur sistem, analisis risiko, serta penyusunan model desain awal untuk menggambarkan bagaimana sistem akan direalisasikan secara teknis.

Dengan demikian, *Elaboration* menjadi fase transisi dari pendekatan konseptual berbasis kebutuhan menuju pendekatan arsitektur yang lebih teknis. Model-model seperti diagram *use case*, *class diagram* awal, *sequence diagram*, dan *sitemaps* biasanya mulai disusun pada tahap ini untuk memperjelas struktur dan perilaku sistem [15].

3. Construction

Berikutnya, tahap *Construction* merupakan fase pengembangan utama, yaitu relaisasi rancangan sistem ke bentuk implementasi teknis. Pada tahap ini dilakukan:

- a. Pengembangan kode program
- b. Perancangan dan implementasi *database*
- c. Pembangunan fitur-fitur sistem
- d. Penyusunan antarmuka pengguna
- e. Integrasi seluruh komponen yang telah dirancang

Tahap ini menghasilkan sistem yang lengkap secara fungsional, meskipun masih memerlukan penyempurnaan dan pengujian lebih lanjut [15].

4. Transition

Tahap *Transition* merupakan fase akhir yang menandakan bahwa sistem telah mencapai tingkat kesiapan untuk digunakan oleh pengguna akhir (*end user*). Pada tahap ini dilakukan pengujian menyeluruh untuk memastikan bahwa sistem berfungsi secara benar, stabil, dan sesuai dengan kebutuhan yang telah

ditentukan. Tahap *Transition* memastikan bahwa sistem dapat diterima dan digunakan secara efektif oleh pengguna [15].

Implementasi metode OOAD ini diharapkan dapat mengatasi permasalahan ketidakterhubungan antara *Web E-Ticketing* dan *Web Back Office* dengan membangun sistem menjadi lebih terstruktur. Pendekatan berorientasi objek pada OOAD ini diharapkan dapat membuat *Web Back Office* dapat menerima data secara langsung dari *Web E-Ticketing*, tanpa harus melakukan input manual, seperti hasil yang dicapai oleh Syah [13]. Selain itu, implementasi metode OOAD ini juga diharapkan menggambarkan visualisasi proses bisnis secara tepat melalui output dari setiap tahapan.

Proses dalam OOAD:

1. Proses Makro

Proses makro merupakan penjelasan keseluruhan proses dari siklus hidup pengembangan sistem yang menyediakan sebuah kerangka kerja untuk proses mikro. Tahapan dalam proses makro dijelaskan sebagai berikut.

a. Requirement

Pada tahap ini, menjelaskan tentang bagaimana membuat dan menjaga SLA dengan *user* dan *stakeholder* mengenai apa yang harus dilakukan dalam pengembangan sistem. Selain itu, tahapan ini juga mendefinisikan batasan-batasan dari suatu sistem yang akan dibuat.

b. Analysis and Design

Tahapan ini menjelaskan bagaimana mengkonversi kebutuhan-kebutuhan sistem yang telah dibuat sebelumnya menjadi sebuah bentuk rancangan sistem, yang mana rancangan tersebut disajikan sebagai spesifikasi dari implementasi sistem dalam memilih lingkungan penerapannya. Selain itu, tahapan analisis dan desain juga mencakup dalam mengembangkan sebuah arsitektur atau bentuk rancangan yang sesuai untuk sebuah sistem integrasi.

c. Implementation

Setelah membuat suatu kebutuhan sistem, menganalisis dan merancang sistem yang akan dibuat. Kemudian, pada tahap ini dilakukan implementasi unit sistem dan menggabungkan rancangan dengan sistem, maksudnya adalah mengimplementasikan rancangan tersebut dalam sebuah sistem atau program dalam bentuk *coding* program.

d. Test

Pada tahap ini, pengujian dilakukan untuk meyakinkan bahwa sistem telah sesuai dengan kebutuhan-kebutuhan yang telah dibuat sebelumnya. Pada prosesnya, perlu dilakukan validasi terhadap fungsi sistem yang telah konkrit.

e. Deployment

Tahap deployment merupakan proses penerapan sistem ke lingkungan operasional. Aktivitasnya meliputi instalasi, konfigurasi, pelatihan pengguna, serta persiapan infrastruktur pendukung agar sistem dapat digunakan secara optimal.

Proses makro memiliki aktivitas-aktivitas pendukung di dalam pembuatan softwaranya, yang mencakup *Project Management*, *Configuration and Change Management*, dan *Environment* [15].

a. Project Management

Manajemen proyek ini dilakukan untuk mengelola proyek pengembangan software, meliputi perencanaan, staffing, dan *monitoring* proyek serta mengatur resiko yang mungkin terjadi.

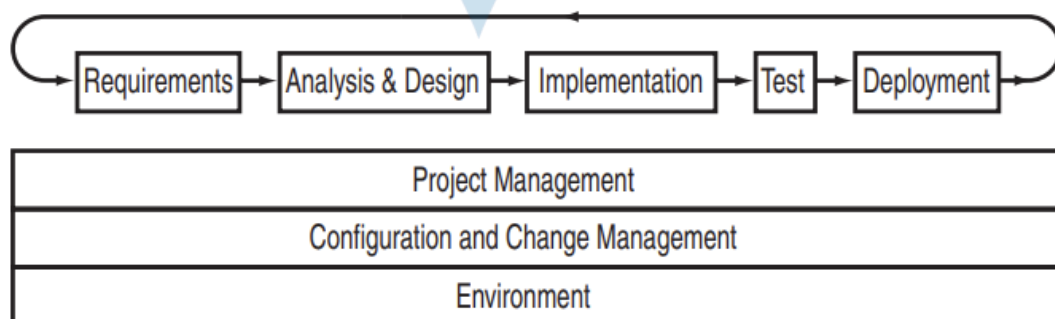
b. Configuration and Change Management

Pada proses ini, dilakukan identifikasi konfigurasi setiap item, mengontrol perubahan pada setiap item, dan mengatur konfigurasi pada setiap item.

c. Environment

Pada proses environment, disediakan sebuah lingkungan pengembangan software, yang meliputi dua proses sebelumnya (*Project Management* dan *Configuration and Change Management*) dan alat-alat yang mendukung pengembangan *software*.

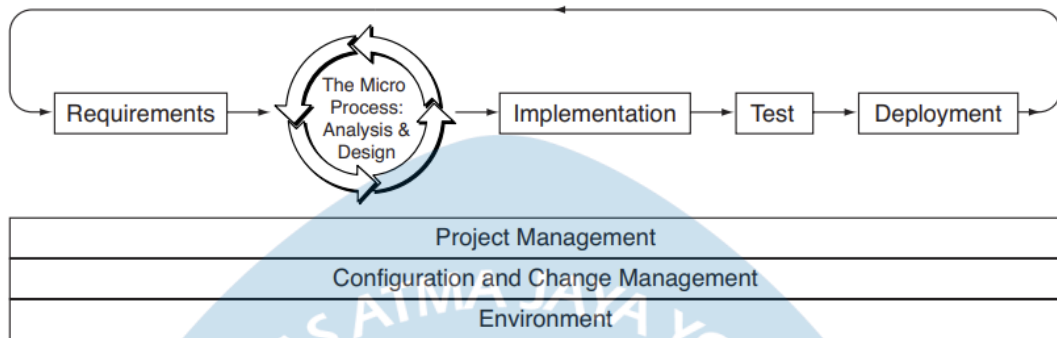
Secara visual, proses makro pada metode OOAD digambarkan dalam Gambar 2.1 berikut.



Gambar 2.1 Tahapan dan Aktivitas Proses Makro [15]

2. Proses Mikro

Selanjutnya, pada tahap mikro, meliputi proses *analysis and design* dengan melihat aktivitas apa yang dilaksanakan dan mengerjakan produk apa yang dihasilkan. Proses mikro merupakan bagian dari proses makro seperti yang terdapat pada Gambar 2.2 berikut.



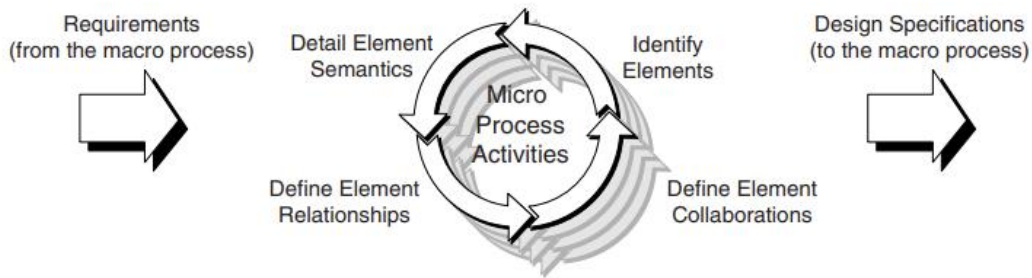
Gambar 2.2 Proses Mikro pada Proses Makro [15]

Seperti yang divisualisasikan dalam gambaran proses mikro tersebut, proses utama dari proses mikro adalah:

- Architecture Description**, pada prosesnya akan menjelaskan arsitektur sistem, termasuk deskripsi mekanisme umum
- Analysis/design model**, pada prosesnya mencakup analisis dan elemen desain Solusi *software*, dan deskripsi bagaimana kebutuhan perilaku sistem yang diwujudkan dalam elemen-elemen.

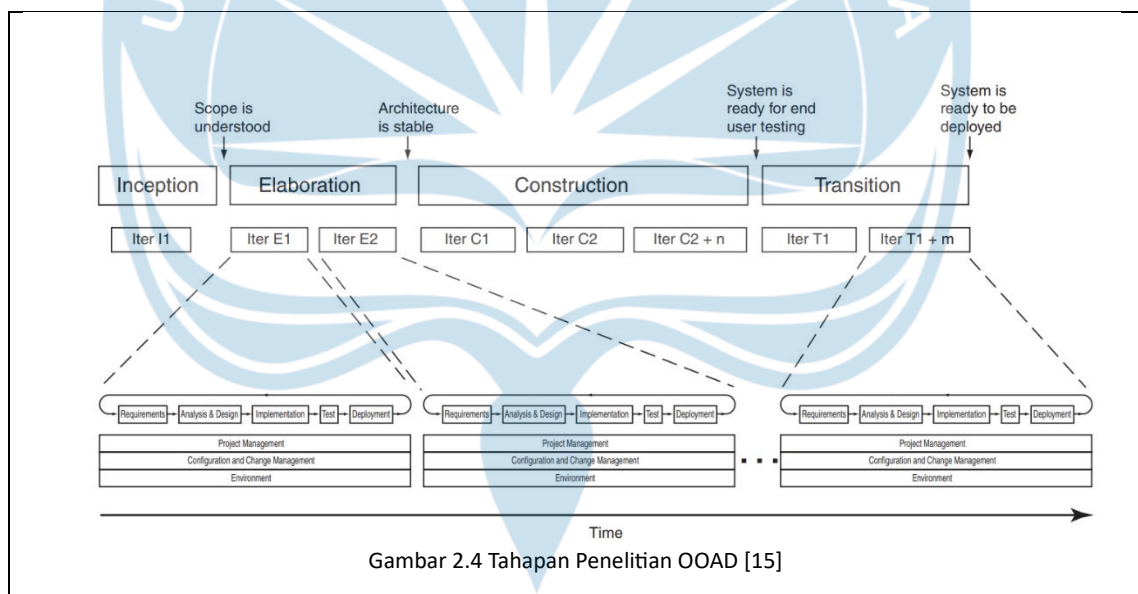
Aktivitas mikro pada prosesnya terdiri dari empat aktivitas seperti yang digambarkan pada Gambar 4, yaitu:

- Identify the elements**, untuk menemukan elemen-elemen yang dikerjakan
- Define the collaborations between the elements**, untuk mendeskripsikan bagaimana mengidentifikasi kerjasama antar elemen untuk menyediakan kebutuhan perilaku sistem
- Define the relationship between the elemen**, untuk mendefinisikan hubungan antar elemen untuk mendukung kerjasama antar elemen
- Define the semantic of the elements**, untuk menciptakan *behaviour* dan *attributes* dari pengidentifikasian elemen.



Gambar 2.3 Aktivitas Proses Mikro [15]

Selanjutnya, aktivitas OOAD dibahas menggunakan Gambar 2.4 berikut. Pada gambar berikut, dijelaskan alur empat tahapan utama dalam metode OOAD, yaitu *Inception*, *Elaboration*, *Construction*, dan *Transition* yang berlangsung secara iteratif dan inkremental. Setiap tahap memiliki fokus yang berbeda, mulai dari pemahaman ruang lingkup, pemantapan arsitektur, pengembangan implementasi, hingga persiapan sistem untuk digunakan oleh *end-user*, namun saling berkesinambungan. Selain itu, gambar 2.4 juga menunjukkan bahwa aktivitas seperti requirement, analisis dan desain, implementasi, serta pengujian berjalan paralel sepanjang siklus.



Gambar 2.4 Tahapan Penelitian OOAD [15]

2.2.5. Application Programming Interface

Application Programming Interface (API) merupakan sekumpulan *calls*, *subroutines*, atau *software interfaces* yang memungkinkan suatu aplikasi untuk menggunakan layanan dan fungsi dari aplikasi lain melalui mekanisme komunikasi antar mesin (*machine-to-machine*) [16]. API berfungsi sebagai jembatan penghubung antar sistem sehingga data dapat ditransfer secara otomatis tanpa campur tangan pengguna. Ini mengindikasikan bahwa, penggunaan API sangat relevan dengan topik penelitian ini, melalui API, data dari

Web E-Ticketing dapat dengan otomatis masuk ke dalam data *Web* Back Office, atau dengan kata lain tanpa campur tangan pengguna yang sebelumnya dilakukan dengan menginput manual melalui *spreadsheet* (*Excel*).

Dalam konteks *web*, dikenal istilah *Web* API, yaitu API yang beroperasi melalui jaringan internet dengan menggunakan protokol HTTP dan format pertukaran data yaitu *Javascript Object Notation* (JSON). Berdasarkan penelitian oleh Novianto, implementasi *RESTful API* memudahkan integrasi dan mendukung komunikasi secara *real-time* [17]. Selain itu, menurut APIs4DGov oleh European Commission, API memiliki peranan strategis dalam mendukung transformasi digital, dalam konteks ini integrasi antara *web* E-Ticketing dan *web* Back Office, yaitu integrasi antar sistem, efisiensi waktu dan tenaga, konsistensi dan akurasi data, serta fleksibilitas dan skalabilitas [16].

Penerapan strategis API telah memberi manfaat dan terbukti efektif dalam penelitian oleh Puspaningrum, yang menggunakan *RESTful API* untuk mengintegrasikan sistem akademik SIAKAD dan SPADA agar tidak terjadi duplikasi data dan untuk meningkatkan efisiensi pengelolaan layanan akademik [7]. Selain di bidang Pendidikan, penelitian oleh Susanto di Bank Mandiri membuktikan bahwa integrasi berbasis API dengan fitur *barcode* dan notifikasi otomatis meningkatkan efisiensi operasional dan mengurangi kesalahan pencatatan manual [18].