



Dependency Mapping and Risk Assessment for Sequencing Interface Migration

Parag Wadehra

Sr. SAP Architect (Contractor) at Amazon Robotics, Memphis, TN.

Abstract

Interface migration sequencing is often based on technical complexity, although migration stability depends on hidden dependencies, architectural readiness, and the consequences for business processes. This study proposes the DIRS model, which integrates dependency mapping, residual risk assessment, and migration priority selection. The model separates migration admissibility from migration sequencing: the dependency map identifies interdependent components, architectural gates prevent the migration of unprepared groups, and business impact determines the order among the admitted clusters. The analytical application demonstrated that the same position in the dependency graph does not imply the same priority. DIRS links dependency information with the migration admission decision and the resulting migration wave while preserving a separate path for components requiring further preparation. The model is intended for SAP landscapes and other enterprise integration environments undergoing phased migration. Its rules prevent the inclusion of unprepared critical interfaces and make the rationale for each decision available for verification before development and production deployment. The proposed scales and thresholds are normative in nature; their application requires documented assessments and adaptation to the conditions of a specific organization.

Keywords: Interface Migration, Dependency Mapping, Architectural Readiness, Residual Risk, Migration Sequencing.

INTRODUCTION

Enterprise interface migration changes the operation of an interconnected chain that includes the provider system, middleware, authentication mechanisms, data owners, and consumer systems (Alonso et al., 2023). Failure of a single link may lead to message incompatibility, transformation errors, access denial, or disruption of subsequent operations. Static analysis identifies dependencies represented in code and configuration but does not fully capture the routes that emerge during the actual execution of requests (Huang et al., 2023). Semantic relationships between components may also persist beyond the conventional call graph (Fávero et al., 2025). Therefore, migration sequencing cannot be determined solely by platform, protocol, or estimated effort.

Without separating migration feasibility from migration priority, a technically convenient interface may be migrated before its dependent consumer, while a business-critical exchange may be included in a migration wave before compatibility and rollback mechanisms have been prepared. Even compatible versions may generate temporary errors when deployment is not properly coordinated (Oumoussa

& Saidi, 2025). A justified sequencing decision therefore requires a model that first identifies mandatory dependencies and the composition of joint migration groups, then assesses architectural readiness, determines residual risk, and only afterward assigns priority with due consideration of business impact.

The aim of this study is to develop and theoretically substantiate an original model for determining the migration sequence of enterprise interfaces based on their classification, dependency map, architectural readiness, residual risk, and business impact. To achieve this aim, the following objectives were set:

- to systematize methods for dependency mapping, change impact analysis, and migration sequencing determination;
- to identify the types of dependencies capable of changing the order of interface migration;
- to formalize five classes of interface workloads and the rules for forming joint migration groups;
- to develop criteria for assessing business impact, architectural readiness, and residual risk;

Citation: Parag Wadehra, "Dependency Mapping and Risk Assessment for Sequencing Interface Migration", Universal Library of Engineering Technology, 2026; 3(4): 07-14. DOI: <https://doi.org/10.70315/uloap.ulete.2026.0304002>.

- to establish rules for blocking unprepared groups and their allocation across three migration waves;
- to determine the limitations of the model and the directions for its subsequent validation.

The research hypothesis is that the joint consideration of business impact and residual risk changes the distribution of admissible groups across migration waves compared with sequencing based solely on the dependency map. Among groups with the same admissibility level, priority is given to prepared interfaces with high business impact and acceptable residual risk, while groups with unresolved critical constraints are not included in the current wave.

The scientific novelty consists in the development of a sequential decision-making procedure in which the dependency map determines migration admissibility, the architectural assessment determines group readiness, the residual risk assessment determines the possibility of including the group in the current wave, and business impact determines the priority among the remaining prepared groups. This sequence transforms disparate information about interfaces into a justified migration order and separates mandatory technical constraints from the criteria for selecting the first and subsequent migration waves.

MATERIALS AND METHODS

The study was conducted as a structured review followed by the development of an original model for determining the sequencing of interface migration. No new empirical dataset was created, and no controlled experiment was conducted.

The search was performed in SpringerLink, MDPI, IEEE Xplore, and arXiv using titles, abstracts, and keywords. Google Scholar was used for additional searching and verification of the final versions of publications. Publications from 2023–2025 were considered. The following queries were used: (“microservice dependency” OR “semantic dependency”) AND (“dependency mapping” OR “architecture recovery”); (“change impact analysis” OR “failure propagation”) AND microservices; (“interface migration” OR “API evolution”) AND (sequencing OR prioritization); (“breaking changes” OR “backward compatibility”) AND (microservices OR API); (“migration graph” OR “migration ordering” OR “version orchestration”). The syntax was adapted to the capabilities of each database.

As a result of the search, 48 sources were initially selected. After merging the results, 7 duplicate records, 6 short abstracts, promotional materials, and publications without accessible full texts, 9 studies unrelated to dependencies, compatibility, or migration sequencing, and 6 preprints for which final journal or conference versions were available were excluded. The final corpus consisted of 20 sources: 15 scientific publications and 5 official SAP documents. The scientific sample included journal articles, full conference papers, review articles, and preprints with transparent descriptions of methods, results, and limitations. The

SAP documents were used only to define Clean Core, SAP Integration Suite, IAS/XSUAA, and the transition tools from SAP Process Orchestration.

Each study was coded for the presence of dependency mapping, change impact analysis, risk propagation assessment, migration sequencing mechanism, and compatibility control. Sequencing studies included publications containing explicit rules for ordering, grouping, precedence, or coordinated deployment. The following indicators were extracted: the number of systems and services, the number of dependent component pairs, F-measure, balanced accuracy, reduction in regression testing volume, workload, and error window duration. The indicators were retained in their original units. No meta-analysis was performed due to differences in architectures and validation methods.

The original model was developed through the sequential identification of five classes of interfaces, mapping of provider and consumer systems, authentication mechanisms, middleware, data owners, and contract versions. Cyclic dependencies were combined into joint migration groups. Business impact, compatibility, security, data status, and operational readiness were then assessed. Groups with unresolved dependencies or critical risk were blocked, while the remaining groups were allocated across three migration waves.

The model was refined taking into account anonymized professional experience. Client architectures, configurations, and operational logs were not used. The internal estimates of 30% and 35% were not used to calibrate the scales or validate the model. The evaluation rules and migration wave assignment rules represent an original normative construct that has not undergone statistical calibration and has not been validated in a controlled experiment.

RESULTS

The quantitative data identified three independent sources of uncertainty in migration planning: incomplete dependency discovery, the discrepancy between structural connectivity and the actual impact scope, and transitional-state errors during the simultaneous operation of different versions. On this basis, the requirements for the DIRS model were formulated, and an analytical validation of its rules was carried out on a synthetic set of interfaces.

The structural dependency map does not cover all relationships capable of affecting interface migration. In the analysis of 27,221 pairs among 41 services, the semantic method achieved an F1 score of 0.813 and a balanced accuracy of 0.944. Additional validation showed that six disputed pairs represented genuine semantic matches (Abdelfattah et al., 2025). Therefore, the absence of a direct call does not exclude a dependency through shared data semantics, consistent state, or the sequence of business operations.

The result of architecture recovery also depends on the selected tool. In a comparison involving 17 applications,

13 static analysis tools were identified, nine of which could be applied in practice. The best individual tool achieved an F1 score of 0.86, whereas combining the results increased the score to 0.91 (Schneider et al., 2025). The difference of 0.05 indicates that information from a single analyzer is insufficient for a final decision on the composition of a migration group. At the same time, combining tools reduces, but does not eliminate, the probability of missed and incorrectly identified relationships.

Execution data refine the actual scope of change impact. In four examined systems, regression test selection based on observed execution paths reduced the number of tests by 40–57%, with an average reduction of 50%, while preserving 100% of the tests that detected failures at the threshold established by the authors (Chen et al., 2023). This result confirms the usefulness of execution logs and tracing for selecting the chains to be verified. It cannot be transferred to unobserved operating modes: the absence of a call within a limited time window does not prove the absence of a dependency.

A separate risk arises during the transition between versions. In a test involving three services under a load of 100 requests per second for ten minutes, sequential deployment created an approximately one-minute error period. During coordinated switching between parallel environments, no errors were recorded (Yaroshynskiy et al., 2025). Compatibility of the final versions therefore does not describe the entire migration risk. The deployment order, the temporary coexistence of instances, consumer readiness, and rollback capability must be verified.

The obtained results defined four requirements for DIRS. The dependency map must combine structural, semantic, and observed relationships. Cyclically dependent interfaces must be treated as a single group. The technical admissibility of migration must be separated from its business priority. Critical residual risk must block the inclusion of a group in a migration wave. The quantitative indicators have different units of measurement and characterize different system properties; therefore, they were not combined into a single performance indicator.

A synthetic dataset consisting of 12 interfaces was created

to validate the rules. It reproduces typical relationships in an enterprise integration environment but does not contain customer architecture information, configurations, logs, or performance indicators. The dataset included five business-critical real-time exchanges, one batch interface, two master data synchronization interfaces, two reporting and analytical interfaces, and two service interfaces.

The interfaces were grouped into seven groups. Group G1 included the authentication and certificate service; G2 included the synchronization of materials and business partners; G3 included order processing, inventory reservation, and production confirmation; G4 included financial posting and settlement confirmation; G5 included operational reporting and analytical data loading; G6 included notifications; G7 included batch archival export.

The dependency map contained four intergroup relationships: G1 preceded G3, G2 preceded G3 and G5, and G3 preceded G4. Within G3, three interfaces formed the closed chain “order – reservation – production confirmation – order.” Splitting such a chain across different migration waves would create an intermediate state with an incomplete business process; therefore, its components were combined into a single joint migration group. This decision is consistent with the principle of grouping interdependent components and with the results of the assessment of uncoordinated deployment of related services (Mens & Decan, 2024).

For the analytical validation, sequential rules were applied, the complete specification of which is presented in Section 4.3. An incomplete mandatory predecessor or a score of 4–5 for at least one risk component assigned a group to the preparation stage regardless of its business impact. The first migration wave included admissible groups with business impact scores of 4–5 and risk scores of 1–2; the second wave included groups with business impact scores of 3–5 and a maximum risk score not exceeding 3. The third wave included groups with business impact scores of 1–2 and a maximum risk score not exceeding 3. A value of 1 indicated the minimum level, whereas a value of 5 indicated the critical level. The established thresholds represent a normative rule of DIRS and are not considered empirically optimal. Table 1 presents the results of applying the model to the synthetic interface dataset.

Table 1. Analytical Application of DIRS to a Synthetic Portfolio of 12 Interfaces

Sources	Cluster; interfaces; class	Required predecessors	Business impact / residual risk	Architecture readiness	DIRS decision
(SAP SE, n.d.-c); (SAP SE, n.d.-d)	G1; 1; V	Her	5 / 1	Ready	Wave 1
(Abdelfattah et al., 2025); (Cerny et al., 2025)	G2; 2; III	Her	4 / 2	Ready	Wave 1
(Loebenberg et al., 2025)	G3; 3; I	G1, G2	5 / 3	Ready with mitigation controls	Wave 2
(Yaroshynskiy et al., 2025)	G4; 2; I	G3	5 / 5	Consumer compatibility and rollback not validated	Pre-Wave Track

(Chen et al., 2023)	G5; 2; IV	G2	3 / 2	Ready	Wave 2
(SAP SE, n.d.-b)	G6; 1; V	Her	1 / 1	Ready	Wave 3
(SAP SE, n.d.-e)	G7; 1; II	Her	3 / 4	Conditionally ready	Pre-Wave Track

Note. Class I – business-critical real-time integrations; Class II – batch interfaces; Class III – master-data synchronization; Class IV – reporting and analytical interfaces; Class V – utility interfaces. Scores and DIRS decisions were developed for the analytical scenario. The cited sources support the considered dependency and risk categories but are not the sources of the assigned scores.

The dependency map assigned G1, G2, G6, and G7 to the same initial admissibility level. Graph-based sequencing did not distinguish among these four groups because none of them had incomplete predecessors. DIRS separated these groups according to business impact and residual risk. G1 and G2 were assigned to the first migration wave, G6 to the third wave, and G7 was assigned to the preparation stage because of a level 4 risk. The service nature of G1 did not reduce its priority because the critical operational group G3 depended on the authentication mechanism. This result demonstrates why the interface class cannot be used as a direct measure of business impact.

Technical sequencing based on increasing estimated implementation effort produced the order G6, G7, G2, G5, G1, G3, and G4. The first two positions were occupied by the low-complexity groups G6 and G7, whereas DIRS assigned G1 and G2 to the first migration wave; the overlap between the first two positions was 0 out of 2 according to the author's calculation. The technical ordering also included G4 in the overall sequence, although consumer compatibility and the rollback procedure remained unverified. DIRS excluded this group until the critical risk had been eliminated.

After completion of the first migration wave, G3 and G5 became admissible. The high business impact of G3 did not allow it to be migrated before G1 and G2, but it ensured priority immediately after the mandatory conditions had been satisfied. Group G5 was assigned to the same wave with lower business impact and lower risk. Group G4 became graphically admissible after the migration of G3 but retained a blocking level 5 risk. Thus, the model separated dependency-based admissibility from authorization for actual migration.

The robustness of the decision was evaluated by sequentially changing each available business impact or risk score by one point within the scale from 1 to 5 while preserving the dependency structure. Of the 21 admissible changes, 16 did not affect the final decision, whereas 5 changed the migration wave or admissibility status. According to the calculation based on Table 1, robustness was 76.2% and local sensitivity was 23.8%. The changes affected groups G2, G3, G5, and G7, whose scores were close to the established thresholds. For G4, reducing the residual risk from 5 to 4 still resulted in assignment to the preparation stage because both levels are blocking. This result characterizes the sensitivity of the rules to boundary scores but does not represent a statistical validation of the model.

In the analytical scenario, the joint consideration of business impact and residual risk changed the distribution of groups compared with technical and graph-based sequencing. Groups belonging to the same initial admissibility level were distributed across the first and third migration waves and the preparation stage, while a group with completed predecessors but critical risk was not admitted for migration. This supports the hypothesis at the level of the internal discriminative capability of DIRS decisions. The scenario does not demonstrate reductions in downtime, migration cost, or the number of production failures. The assessment was performed within a single analytical procedure; therefore, inter-rater agreement and the external reproducibility of the model require separate validation.

DISCUSSION

A static dependency map reflects the relationships that can be recovered from source code, configurations, and contracts, but its completeness depends on the capabilities of the selected tool. A comparison of architecture recovery tools demonstrated differences in the detected relationships and an increase in the F1 score from 0.86 to 0.91 when the results of multiple analyzers were combined (Schneider et al., 2025). Even combined analysis retains the possibility of missing dependencies arising through data, middleware, or the actual sequence of operations.

System execution monitoring refines the execution paths that are actually affected by a change. Selection based on such paths reduced the regression test sets by 40–57% while preserving the tests that detected failures within the established threshold (Chen et al., 2023). However, an observed relationship depends on the observation period and the composition of the recorded workload. An infrequent exchange may be absent from the logs while still remaining mandatory for financial period closing, disaster recovery, or another irregular operation.

Semantic analysis identifies relationships that cannot be inferred from the direct call graph. Two interfaces may use different endpoints and protocols while modifying the same business object or maintaining the consistency of a shared state. During the analysis of semantic dependencies, genuine matches were identified among pairs that had initially been considered disputed (Nicolas-Plata et al., 2024). Therefore, structural, execution-observed, and semantic evidence describe different aspects of the same dependency and should be considered together before migration groups are formed.

Strongly connected components (SCCs) identify interfaces between which cyclic dependencies exist. Splitting such a group creates an intermediate state in which some components expect a new contract or data format, while others continue operating under the previous rules. Combining the cycle into a single cluster eliminates the internal contradiction in migration sequencing. After SCC condensation, the relationships between clusters form a directed acyclic graph (DAG), which defines mandatory predecessors and admissible subsequent steps (Loebenberger et al., 2025).

A DAG establishes technical admissibility but does not rank independent clusters at the same level. If four groups have no incomplete predecessors, the graph considers them equally admissible. However, stopping order processing and migrating an auxiliary export produce different consequences for the enterprise. The structural position also provides no information about certificate readiness, rollback capability, data ownership, or the availability of production monitoring.

Compatibility of the final contract versions does not eliminate the risk of the transitional state. During the sequential update of three related services, errors were observed for approximately one minute, although the target versions were compatible. Coordinated switching between parallel environments was completed without any recorded errors (Yaroshynskiy et al., 2025). Migration sequencing should therefore take into account the deployment strategy and the period during which different versions coexist, rather than only the final system state.

Dependency mapping determines whether a migration step is admissible. Business impact and residual risk determine which admissible step should be performed first.

In the SAP landscape, an interface is admitted for migration only after the operability of the entire exchange chain has been confirmed in the transitional state. An error missed before development affects active integrations after deployment and requires coordinated correction across all related components. Therefore, architectural validation precedes the assignment of a migration wave.

The validation process begins by identifying the business object being exchanged and the organizational unit responsible for it. The data path is then traced from the provider to the consumer, and the current and target contract versions are compared. If simultaneous consumer update is not possible, an adapter or a temporary parallel version is provided. The absence of a confirmed data owner blocks migration because the transformation and recovery rules remain undefined.

After contract validation, the authentication flow is analyzed across the entire request path. The configuration must preserve trust and the required authorizations during the transition, while certificates must remain valid throughout the migration window. SAP Identity Authentication Service

and XSUAA perform different functions in establishing trust and granting access; therefore, their readiness is assessed in the context of the specific integration path (SAP SE, n.d.-c). Migration of dependent interfaces is permitted only after trust has been established between services and identity providers (SAP SE, n.d.-d).

Operational readiness requires verified mechanisms for error handling, retries, logging, auditing, and monitoring. Throughput, high availability, rollback procedures, and assignment of the support team are verified. The presence of a technical mechanism without an assigned owner and a verified operating procedure is not considered readiness: in the event of a failure, the signal source, stopping criterion, responsible specialist, and recoverable system state must be defined in advance.

The Clean Core principle restricts modifications to the ERP core and directs extensions to managed integration points (SAP SE, n.d.-a). In the context of DIRS, this restriction affects architectural admissibility: an interface requiring a new core modification requires redesign or a justified exception. SAP Integration Suite and its Cloud Integration capability provide routing, transformation, monitoring, and integration component reuse (SAP SE, n.d.-b). The use of the platform itself does not automatically reduce risk; the decisive factors remain the specific contracts, error-handling rules, and operational readiness.

Migration from SAP Process Orchestration requires a preliminary assessment of the portability of existing integration scenarios. Automatically portable components, partially supported scenarios, and flows requiring redesign impose different sequencing constraints (SAP SE, n.d.-e). Grouping solely by the source platform may place interfaces with different contractual and operational risks into the same migration wave. DIRS treats the portability assessment result as an input to architectural admissibility rather than as a ready-made priority decision.

The generalization of project experience indicates the practical value of standardizing the integration layer. In one project covering approximately 50 sites, the implementation of a standardized middleware layer was associated, according to an internal assessment, with an approximately 30% reduction in integration costs. In another project, the use of reusable SAP Cloud Integration templates was associated with an approximately 35% improvement in deployment efficiency. These values were obtained retrospectively, were not independently validated experimentally, and were not used to calibrate DIRS.

The DIRS model was developed through the synthesis of the identified requirements and the generalization of cross-project methodology. It sequentially determines the composition of joint migration groups, verifies their technical readiness, and establishes the migration sequence of the admitted groups.

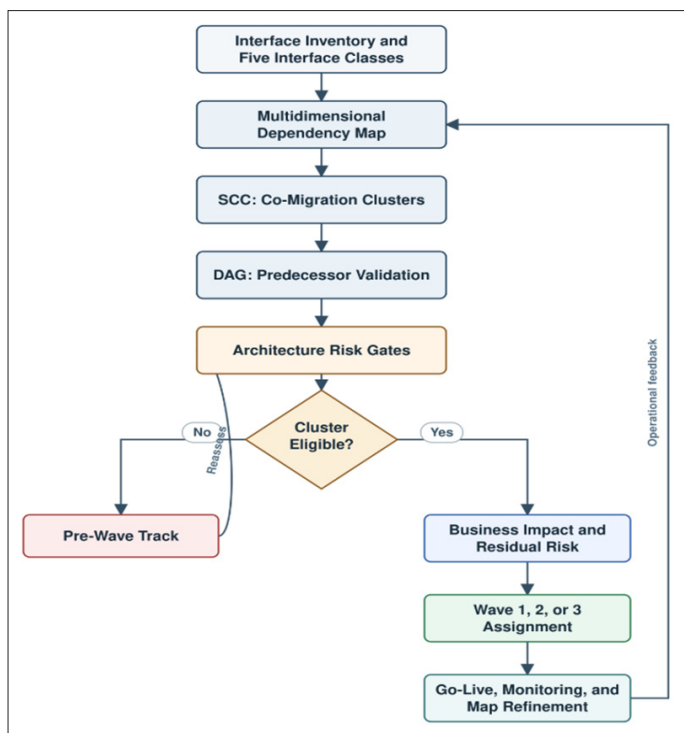


Fig. 1. Migration Sequencing Model Based on Dependencies, Business Impact, and Risk (Author's elaboration)

At the first stage, interfaces are classified into five classes: business-critical real-time exchanges, batch interfaces, master data synchronization, reporting and analytical interfaces, and service interfaces. The class describes the purpose and operating mode but does not replace the assessment of business impact. An authentication service component may receive a higher priority than a reporting interface if multiple operational processes depend on it.

Table 2. Observable Five-Point Scoring Rubric of the DIRS Model

Sources	Criterion	1 – Minimal	2 – Low	3 – Moderate	4 – High	5 – Critical
(Michael Ayas et al., 2023)	Business impact	No process disruption	Local inconvenience	Degraded process; manual workaround	Time-critical or multi-unit disruption	Sales, production, inventory, or finance stopped
(Cerny et al., 2025);	Contract risk	Compatible; tests passed	Additive change; consumers verified	Adapter or dual version required	Breaking change; consumer checks incomplete	Critical consumer incompatible
(Loebenberger et al., 2025)	Dependency risk	No critical external dependencies	1–2 mapped dependencies	3–5 dependencies or shared middleware	6–10 dependencies or cross-domain coupling	More than 10, unresolved cycle, or incomplete map
(SAP SE, n.d.-c); (SAP SE, n.d.-d)	Security risk	Authentication, roles, and trust validated	Minor configuration pending	Non-critical control pending	Certificates, trust, or roles untested	Authentication unavailable or critical control absent
(Abdelfattah et al., 2025); (Yaroshynskiy et al., 2025)	Data and state risk	Stateless or state-compatible	Additive schema change	Transformation required	Synchronization or ownership unresolved	Data-loss risk, corruption risk, or unknown owner
(Chen et al., 2023); (SAP SE, n.d.-b); (SAP SE, n.d.-e)	Operational risk	Monitoring, retry, and rollback validated	Minor control pending	Partial monitoring or rollback untested	Rollback or high availability unverified	No monitoring, recovery path, or operational owner

Note. The scales and level boundaries were developed in this study. The cited sources support the criteria but do not establish the empirical optimality of the thresholds.

The multidimensional map records providers and consumers, authentication methods, certificates, middleware, protocols, data owners, state management, contract versions, observed execution paths, semantic relationships, and external platforms. Relationships are confirmed using multiple sources because static analysis, execution logs, and semantic matching have different limitations (Alhanahnah et al., 2024). A change affecting a recorded request chain refines the impact scope but does not eliminate the need to verify infrequent and event-dependent execution paths (Hassan et al., 2025).

The identified cycles are condensed into SCCs and form joint migration groups. After SCC condensation, the graph is transformed into a DAG, in which mandatory predecessors are verified (Loebenberger et al., 2025). An incomplete predecessor means that the cluster is not yet admitted to a migration wave. This separation prevents high business priority from being used to bypass a technical dependency.

After graph-based validation, the cluster passes through architectural gates. Contract and version compatibility, dependency completeness, authentication, authorization, certificates, data ownership, state synchronization, error handling, retries, monitoring, auditing, rollback, performance, high availability, Clean Core compliance, and operational owner readiness are assessed. Any unmet gate is translated into a residual risk that must be eliminated or accepted according to the established procedure.

The assessment is based on observable evidence. Business impact and each risk component are assigned a score from 1 to 5. The complete scoring rubric is presented in Table 2.

The overall residual risk level is determined by the highest score among contract risk, dependency risk, security, data and state, and operational readiness. The maximum value preserves the visibility of a critical constraint. Averaging could conceal a single unacceptable risk behind several low scores: a combination of four scores of 1 and one score of 5 does not constitute a safe authorization for migration.

A cluster is assigned to the preparation stage if at least one mandatory predecessor has not been completed or if any risk component receives a score of 4–5. The same decision is made when there is no assigned data owner or support owner, no compatible contract, no mechanism for version coexistence, or no prepared rollback procedure for a business-critical interface. High business impact does not override the blocking condition.

The preparation stage includes feasibility assessment, dependency refinement, contract adaptation, risk mitigation, redesign, or evaluation of whether the interface should be decommissioned. After the corrective action has been completed, the cluster passes through the architectural gates again. Assignment to a migration wave is permitted only after all risk components have been documented as reduced to an acceptable level.

The first migration wave includes clusters with business impact scores of 4–5, risk scores of 1–2, and completed predecessors. The second migration wave includes admissible clusters with business impact scores of 3–5, provided that the highest residual risk does not exceed 3. The third migration wave is intended for clusters with business impact scores of 1–2 and risk scores not exceeding 3. Any risk score of 4–5 or any incomplete predecessor assigns the cluster to the preparation stage.

If multiple clusters qualify for the same migration wave, priority is given to the cluster affecting the larger number of downstream consumers. If the scope is equal, preference is given to the group with higher operational readiness. The final criterion is the ability of the cluster to establish the technical foundation for subsequent migration waves, for example by preparing a shared authentication mechanism, an integration route, or monitoring capabilities.

The boundary scenarios illustrate the internal logic of the rules. A cluster with a business impact score of 5 and all risk scores of 1–2 is assigned to the first migration wave. With the same business impact but a security risk score of 4, it is assigned to the preparation stage. A combination of a business impact score of 3 and a maximum risk score of 3 corresponds to the second migration wave, whereas a business impact score of 2 and a maximum risk score of 2 corresponds to the third migration wave. These scenarios are illustrative and do not contain customer data. They verify the internal consistency of migration wave assignment rather than the operational effectiveness of the model.

The scoring rubric and the migration wave assignment rules

represent an initial normative decision-making framework proposed by the author. They have not yet undergone statistical calibration; therefore, the established thresholds should not be interpreted as empirically optimal values. In a specific organization, their application requires uniform rules for documenting assessments, identifying the source of each piece of evidence, and re-evaluating boundary cases.

The analytical validation supports the internal consistency of the proposed rules. Sequencing based solely on dependencies does not distinguish clusters belonging to the same admissibility level; DIRS distributes them according to business impact and residual risk while simultaneously blocking any critical constraint regardless of business significance. The empirical effectiveness of the model remains a subject for future evaluation.

CONCLUSION

Interface migration sequencing should be established only after technical admissibility has been verified, rather than replacing it. An incomplete dependency or a level 4–5 risk excludes a cluster from the current migration wave regardless of its business impact.

DIRS makes the basis of each decision verifiable: the migration wave is determined by established dependencies, architectural readiness, and documented residual risk. The preparation stage preserves process controllability because a blocked cluster is returned for reassessment after the specific constraint has been eliminated.

The analytical scenario demonstrated the ability of the rules to distinguish clusters belonging to the same graph level and to modify technical sequencing by taking business consequences into account. The proposed scales and thresholds remain normative. Their application requires uniform documentation criteria and subsequent validation within a specific integration landscape.

REFERENCES

1. Abdelfattah, A. S., Cordes, K. E., Medina, A., & Cerny, T. (2025). Semantic dependency in microservice architecture: A framework for definition and detection. arXiv. <https://doi.org/10.48550/arXiv.2501.11787>
2. Alhanahnah, M., Boshmaf, Y., & Baudry, B. (2024). DepsRAG: Towards managing software dependencies using large language models. arXiv. <https://doi.org/10.48550/arXiv.2405.20455>
3. Alonso, J., Orue-Echevarria, L., Casola, V., et al. (2023). Understanding the challenges and novel architectural models of multi-cloud native applications: A systematic literature review. *Journal of Cloud Computing*, 12, 6. <https://doi.org/10.1186/s13677-022-00367-6>
4. Cerny, T., Goulis, G., & Abdelfattah, A. S. (2025). Towards change impact analysis in microservices-based system evolution. In *Proceedings of the 2025 IEEE International*

- Conference on Software Analysis, Evolution and Reengineering (SANER) (pp. 159–169). IEEE. <https://doi.org/10.1109/SANER64311.2025.00023>
5. Chen, L., Wu, J., Yang, H., et al. (2023). A microservice regression testing selection approach based on belief propagation. *Journal of Cloud Computing*, 12, 20. <https://doi.org/10.1186/s13677-023-00398-7>
6. Fávero, L. F., Almeida, N. R. d., & Affonso, F. J. (2025). A systematic mapping study on the modernization of legacy systems to microservice architecture. *Applied System Innovation*, 8(4), 86. <https://doi.org/10.3390/asi8040086>
7. Hassan, H., Abdel-Fattah, M. A., & Mohamed, W. (2025). A pattern-based framework for automated migration of monolithic applications to microservices. *Big Data and Cognitive Computing*, 9(10), 253. <https://doi.org/10.3390/bdcc9100253>
8. Huang, H.-Y., Fanjiang, Y.-Y., Hung, C.-H., Tsai, H.-Y., & Lin, B.-H. (2023). Evaluation of a smart intercom microservice system based on the Cloud of Things. *Electronics*, 12(11), 2406. <https://doi.org/10.3390/electronics12112406>
9. Loebenberger, D., Hirsch, E., Gazdag, S.-L., Herzinger, D., Näther, C., & Steghöfer, J.-P. (2025). On the formalization of cryptographic migration. *arXiv*. <https://doi.org/10.48550/arXiv.2408.05997>
10. Mens, T., & Decan, A. (2024). An overview and catalogue of dependency challenges in open source software package registries. In *Proceedings of the 23rd Belgium-Netherlands Software Evolution Workshop (BENEVOL 2024)*. <https://doi.org/10.48550/arXiv.2409.18884>
11. Michael Ayas, H., Leitner, P., & Hebig, R. (2023). An empirical study of the systemic and technical migration towards microservices. *Empirical Software Engineering*, 28, 85. <https://doi.org/10.1007/s10664-023-10308-9>
12. Nicolas-Plata, A., Gonzalez-Compean, J. L., & Sosa-Sosa, V. J. (2024). A service mesh approach to integrate processing patterns into microservices applications. *Cluster Computing*, 27, 7417–7438. <https://doi.org/10.1007/s10586-024-04342-5>
13. Oumoussa, I., & Saidi, R. (2025). The ontology-based mapping of microservice identification approaches: A systematic study of migration strategies from monolithic to microservice architectures. *Computers*, 14(4), 133. <https://doi.org/10.3390/computers14040133>
14. Schneider, S., Bakhtin, A., Li, X., et al. (2025). Comparison of static analysis architecture recovery tools for microservice applications. *Empirical Software Engineering*, 30, 128. <https://doi.org/10.1007/s10664-025-10686-2>
15. Yaroshynskyi, M., Puchko, I., Prymushko, A., Kravtsov, H., & Artemchuk, V. (2025). Investigating the evolution of resilient microservice architectures: A compatibility-driven version orchestration approach. *Digital*, 5(3), 27. <https://doi.org/10.3390/digital5030027>
16. SAP SE. (n.d.-a). Clean core integration. SAP Help Portal. Retrieved July 21, 2026, from <https://help.sap.com/docs/cloud-alm/applicationhelp/clean-core-integration>
17. SAP SE. (n.d.-b). Establish trust and federation between SAP Authorization and Trust Management service and SAP Cloud Identity Services—Identity Authentication. SAP Help Portal. Retrieved July 21, 2026, from <https://help.sap.com/docs/btp/sap-business-technology-platform/establish-trust-and-federation-between-uaa-and-identity-authentication>
18. SAP SE. (n.d.-c). SAP BTP security recommendations. SAP Help Portal. Retrieved July 21, 2026, from <https://help.sap.com/docs/btp/sap-btp-security-recommendations-c8a9bb59fe624f0981efa0eff2497d7d/sap-btp-security-recommendations>
19. SAP SE. (n.d.-d). Transition to SAP Integration Suite. SAP Help Portal. Retrieved July 21, 2026, from <https://help.sap.com/docs/migration-guide-po/migration-guide-for-sap-process-orchestration/transition-to-sap-integration-suite>
20. SAP SE. (n.d.-e). What is SAP Integration Suite? SAP Help Portal. Retrieved July 21, 2026, from <https://help.sap.com/docs/integration-suite/sap-integration-suite/what-is-sap-integration-suite>