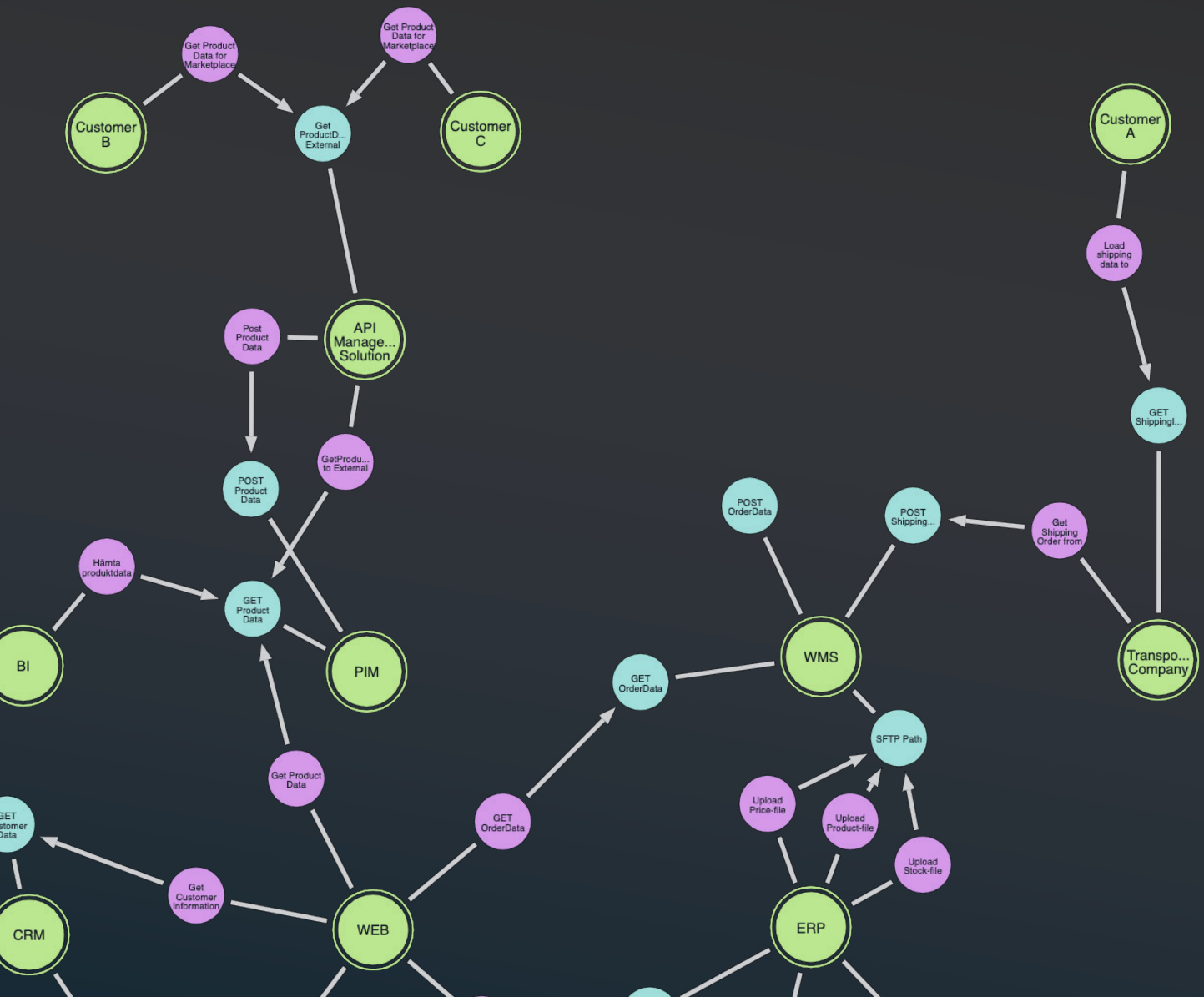
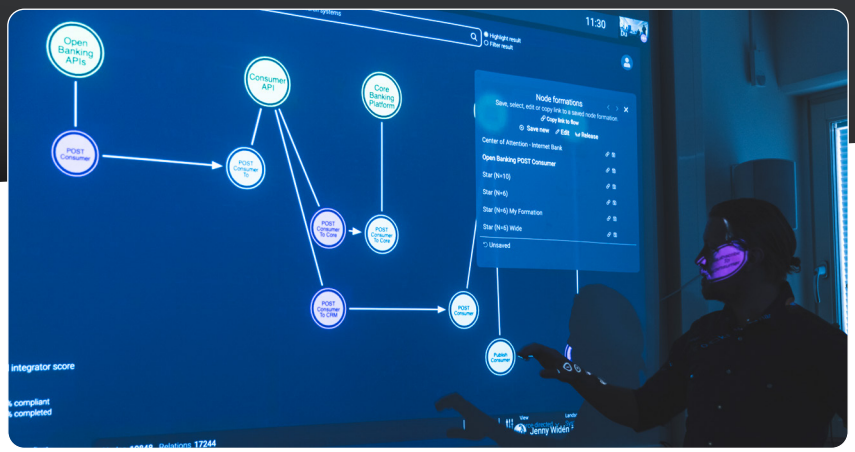
Starlify

15 steps for building better integrations & a durable landscape

Learn the keystones of smart integration



Introduction



In this age of digitalization, the number of applications and systems are growing continuously in most organizations. With increased demands to access the right information at the right time, connecting applications via integrations has become essential for companies to keep up.

This guide summarizes the 15 keystones of **Certified Integrator**, a standardized integration method developed and practiced by Entiros, one of the leading integration companies in Sweden.

Follow these 15 steps to ensure quality and control over your integration landscape and reduce time spent in integration projects by 50 %.

10 essentials for an end-to-end integration

1

Identification – with Registry

An integration should be identifiable with a unique ID for incident, problem, change, and reuse management. Identification should be self-contained in the deployable elements of the integration, and referenced from documentation and other metadata about the integration.

2

Patterns – with Contracts & Services

An integration should be identifiable with a unique ID for incident, problem, change, and reuse management. Identification should be self-contained in the deployable elements of the integration and referenced from documentation and other metadata about the integration.

3

Information – with Requirements Coordination

An integration should be designed with its data object at rest and in transition. Each information object should be described by its entities with name and description. Each entity should be described with relevant details.

4

Mediation – with Test Coordination

An integration should be designed with mediation of information objects with mappings between fields and any operation applied in the mapping. In addition, a mediation sequence diagram should be provided.

5

Scaling – with Capacity Testing

An integration should be designed with performance and capacity limits. The measures that can be taken to scale the integration should be added as change cases and solutions to the design.

Starlify

6

Security – with CIA Ensured

An integration should be designed to balance the security CIA triad – confidentiality, integrity, and accessibility. The design should be made with reference to where in a simplified OSI model (network, transport, message level) each of the three requirements are met, and which measures and responsibilities each party in the integration will meet.

7

Reliability – with Reliability Options

An integration should be designed with a specified level of reliability with respect to its availability, whether it achieves the same result every time, message behavior, and data persistence.

8

Operability – with Trace & Error Handling

An integration should be designed to allow for error handling in the event of non-expected execution. This should include traceability with respect to the integration activity and messages processed by the integration. The integration should have features that allow automated and manual error handling. This integration should allow for external monitoring, detecting non-activity, and error situations.

9

Future-proofing – with Lifecycle Management & Versioning

An integration should be designed with self-contained versioning and life-cycle management with respect to non-breaking changes where the consumer only needs to change their integration use when a major version is introduced.

10

Integration Documentation - Repository

An integration should be documented with respect to the above design, have self-contained code documentation, and have consumer-retrievable documentation specifying how to use the integration.

5 essentials for a better integration landscape

1

Ownership & Audiences

An integration should be designed with a clear understanding and specification of the current and future audience and ownership. Who will benefit from the integration and who's responsible for its lifecycle.

2

Reuse & Architecture

An integration should be designed to maximize reuse in the landscape and extendibility. Growing the integration stakeholder base and reusing the integration in a new version or a separate integration. The purpose and place of the integration in an overall architecture should be noted.

3

Discovery & Governance

An integration should be categorized and cataloged by its purpose to allow stakeholders to discover and use it. The user adoption of an integration should be governed by a user-targeted description of its meaning.

4

Marketing & Onboarding

An integration should be promoted proactively to increase user adoption. Onboarding should be easy by providing a technical manual, onboarding instructions, examples, and other relevant materials. The integration should have a clear product roadmap stating what it does and when new features and enhancements will be added.

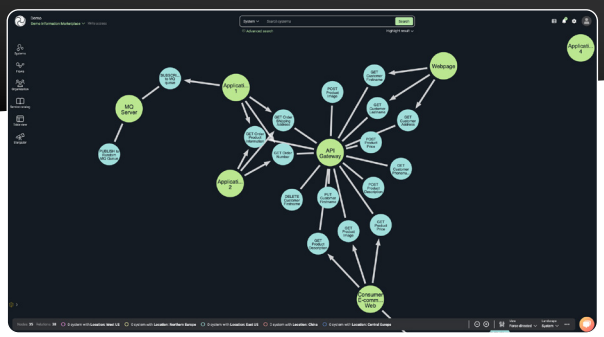
5

Landscape Documentation

An integration should be added to or related to the documentation of its larger context to make landscape stakeholders aware of its existence.

Starlify

Accelerate your integration delivery with Starlify



Easy to follow the standard in Starlify

Starlify is a collaboration tool for integration development that provides exceptional insight into your company's system integration assets by collecting them all in one place.

The 15 keystones described in this guide are incorporated into the Starlify workflow to make life easier for you and your developing teams. Use Starlify and follow the method to ensure quality integrations equipped for the future!

Organize for insight

Organize your integration assets in open, visualized networks, and gain valuable insights.

Go faster

Collect all integration assets in Starlify and make it easy for everyone to follow best practice.

Time to innovate

Spend less time on integration, and focus on what matters. Stop fire fighting and start innovating.

Read more on www.starlify.com

Starlify