FADDOM

The Complete Guide to Application Dependency Mapping

Table of Contents

What is Dependency Mapping?	3
What is Application Mapping (ADM)?	3
Benefits of Application Mapping for Your Organization	4
How Application Dependency Mapping Helps IT Professionals	4
Here are the main ways ADM makes life easier for IT teams:	4
Application Mapping vs. Software Mapping	5
ADM Methods and their Pros and Cons	5
Application MappingTechnologies	6
ADM Use Cases	7
How to Choose the Right ADM Tool	9
ADM In-Depth: Understand Application Mapping Inside Out	10
Categories of Application Dependency	10
Application DependencyMapping Challenges	11
A Dependency Mapping Example	13
ADM Best Practices	14
Keep Tabs on Open-Source Dependencies	14
Minimize Use of Proprietary Dependencies	15
Make Dependencies More Visible and Easier to Discover	15
Take Different Types of Dependencies into Account	15
Faddom's Application Dependency Mapping Software	16
Learn More About Application Mapping	16
See Our Additional Guides on Key Hybrid Cloud Topics	17

What is Dependency Mapping?

Dependency mapping is the process of gathering information about all your underlying application dependencies and presenting it in a way you can easily understand it.

This can be in the shape of a manual spreadsheet, listing details about the different services your application uses and the connections between them. However, in view of the growing complexity of their IT systems, organizations are increasingly using specialist software to map their dependencies.

Such tools provide information not only in spreadsheet form, but also visual representations of entire application ecosystems.

What is Application Mapping (ADM)?

In a complex IT environment, application mapping or network discovery tools, also known as application dependency mapping (ADM), are critical. Application mapping can identify and map out all the instances, communications channels, and applications that are being used in your IT ecosystem as well as the ports and services that are being used. The best solutions can also quickly and easily define VPCs, subnets, and security groups on cloud providers such as AWS, Azure, and GPS.

When displayed on an intuitive map, you have a visual representation of your application dependencies that can be shared, examined, or used for planning and troubleshooting. This visualization can be used for business strategy, organizing by business context, and prioritizing critical alerts and information in real time.

In this article:

- Why is Application Dependency Mapping Important?
- 5 Benefits of Application Dependency Mapping
- 4 ADM Methods and their Pros and Cons
- Application Mapping Technologies
- ADM Use Cases
- How to Choose the Right ADM Tool
- ADM In-Depth: Understand Application Mapping Inside Out
- Faddom ADM: The Answer to IT Complexity (This is part of an extensive series of guides about hybrid cloud.)

Benefits of Application Mapping for Your Organization

Enhanced Visibility and Better Risk Management

IT mapping offers improved visibility and understanding of your IT landscape, enabling strategic decision making by identifying gaps, redundancies, and optimization opportunities. It also helps understand the impact of changes across the IT environment, prevents disruptions, and facilitates communication between IT and business units. Moreover, IT mapping leads to better risk management by identifying potential vulnerabilities and critical dependencies, enabling proper risk mitigation and compliance management.

Improved IT Governance and Cost Optimization

IT mapping promotes effective IT governance by aligning IT investments and activities to business objectives, enhancing project management and IT service management (ITSM). This alignment ensures that IT services contribute to achieving business goals and deliver value. Additionally, IT mapping allows for identifying redundancies and inefficiencies, resulting in cost reduction and optimization.

How Application Dependency Mapping Helps IT Professionals

Here are the main ways ADM makes life easier for IT teams:

1. Respond to problems more rapidly: Through a raft of features, such as the ability to view versions of your maps before an issue appeared, you'll be better informed on how to deal with problems as and when they arise.
2. Gain better service visibility: By providing IT operations management teams with a useful reference point for optimizing resources, diagnosing and resolving issues, tackling security incidents, and maintaining your catalog of technology services.
3. Stay on top of dependency changes: By continually monitoring your application's inventory of dependencies, ADM gives you the confidence that your maps are always up to date and complete.
4. Pinpoint performance bottlenecks: Reducing the time it takes to get to the root cause of many application performance problems, such as slow SQL queries and poorly configured or under-resourced dependencies.
5. Avoid dependencies in the first place: By developing more agnostic applications that are more loosely coupled to dependencies—especially to specific versions of software and operating systems. This will improve the stability of your software and make it more tolerant to issues and changes elsewhere in the application stack.

Learn more about leveraging ADM in your organization:

- Building a change management process
- Using cloud migration tools
- Isolating critical assets with microsegmentation

Application Mapping vs. Software Mapping

While the terms 'application mapping' and 'software mapping' are often used interchangeably, they refer to different aspects of IT mapping. Application mapping is focused on understanding how software applications are interconnected and how they support business processes. It involves identifying the different applications, their dependencies, and how data flows between them. On the other hand, software mapping is focused on the software components themselves. It involves identifying the different software components, their versions, configurations, and how they interact with each other. Software mapping is crucial for managing software assets, ensuring compatibility, and planning for upgrades. Both application mapping and software mapping are important components of IT mapping. They complement each other and together provide a comprehensive view of the software landscape of an organization.

ADM Methods and their Pros and Cons

Here are the primary ways ADM tools discover applications in your environment.

1. Sweep and Poll

How it works: Identifies dependencies by pinging IP addresses and gathering information from the responses.

Pros: Lightweight discovery method, which is comparatively straightforward to perform, as it's able to scan an entire network from a single location.

Cons: Slow for large organizations with large and complex application deployments. Critical assets can potentially go undiscovered in dynamic IT environments, such as the public cloud, because of the time it takes to report changes to network topologies.

2. Network Monitoring

How it works: Uses network flow or packet analysis to identify the path taken by data as it moves through an IT system.

Pros: It is usually a very lightweight deployment, where you can map the entire environment quickly. Detects changes to application topologies in real-time. More effective at discovering dependencies in dynamic cloud-based environments and where existing knowledge of an application ecosystem is less well known. It has a minimal performance overhead on the topology, if any. It is considered the most secure since it is non-intrusive. There is no need to install agents on the machines; it doesn't require credentials to access the servers or to reconfigure firewall rules. The simplest to use with highly segmented environments.

Cons: Lacking in-depth information on the machines since it is non-intrusive – unless you provide credentials to the servers. Information such as installed software and its version, the running processes, and performance data. Can't actively change the configuration of the machines or the network devices.

3. Agent on Server

How it works: Installs agents on servers to monitor inbound and outbound traffic in real time.

Pros: Low bandwidth. Real-time capture of changes to application topologies. Able to differentiate between applications running on the same server with the same IP address. Correspondingly able to provide more detailed application-level insights.

Cons: Agents need to be installed on every configuration item, making it difficult to achieve blanket visibility, especially in large-scale enterprise environments. Moreover, installation and maintenance of agent software can become notoriously complex and time-consuming at scale. The user needs to have a prior understanding of an application topology so they know where to strategically install agents. And if an application processes highly regulated or sensitive data then third-party agents may breach compliance or security requirements.

4. Orchestration-Level Discovery

How it works: Leverages the monitoring capabilities of other technologies to keep track of application components and underlying server resources. For example, some ADM solutions gather data from DevOps tools such as Kubernetes, which manages all underlying application components. Other ADM offerings integrate with APM tools, such as Datadog.

Pros: May help keep resource consumption down by leveraging technologies already in use. Able to consolidate information from different tools to provide more comprehensive insights.

Cons: Reliance on other technologies. Consequently, it can be difficult to gain full visibility over the application ecosystem, as the third-party technology needs to be installed on every single server. This, in turn, can make it a particularly cumbersome and expensive solution.

Application Mapping Technologies

ADM tools set themselves apart through not only different methods of discovering dependencies, but also different methods of organizing and mapping them. A specific solution may use one or more of the following mapping approaches. Pattern-Based Mapping Pattern-based mapping uses scripts that follow a sequence of operations to work out the attributes of application components and their connections. It is a resource-heavy mapping technique, but generates an accurate and comprehensive representation of an application topology.

Traffic-Based Mapping

Traffic-based mapping uses network flow logs to collect and analyze traffic-related data. It is designed to complement pattern-based mapping by casting a finer net to identify components that might otherwise go undetected by other methods. However, it also tends to map configuration items that do not affect the operation of an application.

Tag-Based Mapping

An approach that builds up a picture of the application supply chain from tags assigned to IT assets. It is relatively straightforward to set up and configure compared with other methods and can be useful for mapping applications hosted in virtualized, hyper-converged, and multi-cloud environments—where tagging is standard practice. However, tag-based mapping can produce unsatisfactory results if assets are tagged incorrectly.

AI Mapping

A service mapping process that uses artificial intelligence (AI) to rank application fingerprints based on the extent to which they appear to be bound to that specific application. For example, an internal application component would rank highly whereas a server used for general system monitoring would be given a low relevancy score. AI mapping may be useful for complex application topologies. However, it takes time to train the machine learning algorithms in order to achieve the best results.

ADM Use Cases

IT Change Management

ADM can play a useful role in a wide variety of change management use cases. For example, when teams need to:

- launch new applications or services
- make changes to existing applications — to add new functionality, fix issues, or meet new requirements
- provision or procure new infrastructure, such as storage, routers, and servers increase network visibility
- perform software patches and updates
- do IT service mapping
- make configuration changes to libraries and frameworks, operating systems, and database magement systems
- integrate DevOps and network mapping tools into software development workflows

Compliance

ADM can help you identify gaps in your compliance undertakings.

For example, under the EU General Data Protection Regulation (GDPR), you can only generally process and store personal data about European citizens in a data center location within the European Economic Area (EEA) and a limited number of other permitted countries. You can therefore use ADM to determine where your application data resides and check whether it meets applicable data residency requirements accordingly.

Some ADM tools also provide you with an audit trail of the changing topology of your system. You can use this to demonstrate you take appropriate accountability measures to meet compliance.

Data Center and Cloud Migration

Migration projects are particularly risky, as so many things can potentially go wrong.

ADM can support your preparations by helping you to:

- understand your existing application architecture
- determine what data you need to move
- identify what you need to back up before the migration
- formulate a plan of action
- ensure nothing will be overlooked during the migration process

Cloud Cost Optimization

The public cloud is pay-as-you-go (PAYG) infrastructure where charges are based on your resource usage. This comes at the risk of skyrocketing bills if you're not optimizing your workloads.

ADM provides a clearer understanding of your application deployments, letting you make smarter choices about instance sizing, resource allocation of containers, make better use of provisioned storage, and better match RI purchases with the resource requirements of your applications. ADM also gives you an overview into which application components are consuming the most resources

Cybersecurity

ADM solutions provide a range of insights to help you manage cybersecurity. For example, they are typically able to detect all your SSL/TLS certificates and give you details such as expiry dates so you know when you need to update them.

Furthermore, you can use the network mapping capabilities of ADM to devise a suitable microsegmentation strategy.

This is a security approach where you effectively compartmentalize workloads by defining granular network controls over traffic between the different resources within your application environments. This helps to contain a breach, as it limits an attacker's ability to move laterally across your network.

Business Continuity and Disaster Recovery

Failover systems are notoriously complex. Similarly, backup and recovery systems must be properly synchronized and take into account all of the data applications need.

ADM can help the recovery process—by helping you to determine the order in which you should restore your systems. For example, authentication services would likely be high on your list of priorities so users can log straight back into your applications as soon as they become available again.

ADM can help you draw up a communications plan that will ensure everyone is in the loop during the crisis. ADM also continually monitors your applications for changes. So you'll always have an up-to-date picture of your application topology when you periodically review and test your DR plans.

How to Choose the Right ADM Tool

Different companies have different needs. And, this is equally so when it comes to choosing an application dependency mapping tool. To ensure it will be fit for purpose, it's therefore important to weigh up a whole variety of different factors. These will typically include:

- **Flexibility:** For example, the software should be able to integrate with a wide range of on-premises and cloud-based environments. This helps ensure it's not only suited to existing hardware but also to future infrastructure requirements.
- **Range of features:** Capabilities may include alerting mechanisms, cloud-cost optimization tools to help keep monthly bills down, and varying levels of customization and automation.
- **Pricing:** Each vendor has its own unique pricing structure and licensing options, so it can be difficult to compare products on a like-for-like basis. Prospective customers can, at least, make this easier by having a clear idea of how many servers they want to map at the outset.
- **Visual presentation:** Some solutions offer highly detailed insights, geared towards users with a high level of IT expertise. Others provide much more streamlined information for those who only need a general overview. But, in either case, the visual presentation should still be clear and uncluttered, and ideally complemented by useful contextual information.
- **Ease of use:** The product should be quick and easy to set up, be intuitive to use, and deliver insights that are simple to understand. Many vendors offer a free trial, in most cases for either 14 or 30 days, so users can put offerings through their paces before they buy.

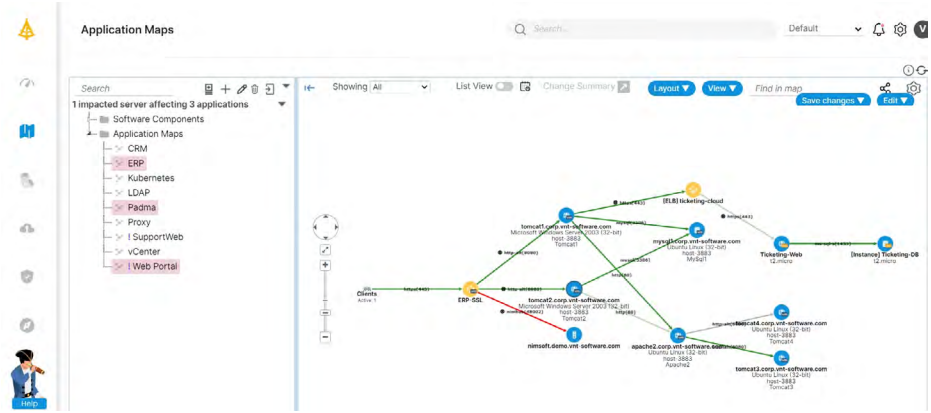
Faddom ADM: The Answer to IT Complexity

ADM solutions support critical processes like incident management, BCDR planning and processes, compliance, cost optimization, and cybersecurity, providing the insights IT teams need.

Faddom is a leading ADM solution that helps organizations make better decisions on how to modernize and manage applications, deal with issues more quickly and efficiently, and even avoid using problematic dependencies in the first place.

Faddom helps companies worldwide with our agentless software that maps entire hybrid IT environments — both in the cloud and on premise — in as little as one hour.

Start a free trial today! >>



ADM In-Depth: Understand Application Mapping Inside Out

If you are planning to deploy ADM in your environment, it can be valuable to understand the application mapping process in more depth. Read on to understand the details behind ADM solutions and better understand how to deploy and leverage them in your organization.

Categories of Application Dependency

Application-to-Application Dependencies

Application-to-application dependencies provide the core functionality of any piece of business software. Such dependencies can be proprietary software or other in-house applications.

Mapping application-to-application dependencies helps IT teams to:

- track the movement of data between different application components
- identify what applications should be migrated or transitioned together
- understand the potential impact of infrastructure changes and plan accordingly
- ensure any BCDR strategy takes the entire functionality of an application into account

It can also prove particularly useful to developers who are unfamiliar with the codebase, giving them a quick and efficient way to identify the basic building blocks of an application and help them understand the potential impact of coding changes.

Service Dependencies

As with application-to-application dependencies, service dependencies support an application, but don't directly affect its functionality. For example, a domain name system (DNS) supports an application by servicing requests to map hostnames to their IP addresses. Service dependencies can also depend on other services, forming a chain of dependencies as part of the application stack. At the same time, some types of service dependency are specifically designed to help overcome issues caused by dependency chains. For instance, message queue services, such as RabbitMQ and Amazon Simple Queue Service (SQS), allow applications to continue processing requests whenever a component within the chain is unreachable, busy, or offline. Service dependencies are often reusable components that can be used by many different types of application. As a result, many of them are open source, as they save organizations the time and money

involved in building their service dependencies from scratch. However, this also comes with a number of implications for dependency management. For example, organizations will need to keep track of such dependencies to ensure they:

- comply with the licensing terms of the open-source software
- fix open-source vulnerabilities by applying the latest patches and software updates
- avoid compatibility issues between the core application and open-source components and between the open-source components themselves

Beware: Service dependencies are often misappropriated in application delivery and migration processes, leading to rogue connections between production and development environments. However, dependency mapping tools help users identify such misconfigurations and prevent any associated security and compliance issues accordingly.

Host-Based Dependencies

Host-based dependencies are those between the application and the underlying host environment. They include:

- physical servers
- virtual machines
- containers
- operating systems
- networks

These dependencies are the foundations upon which applications are supported. Likewise, they provide the framework around which dependencies are mapped by ADM tools. And this makes sense when you consider that an application dependency map is basically a visual representation of the different servers used by an application and the network connections between them.

This is the starting point from which users can do everything else with ADM tools, such as investigating application topologies further, checking the health of servers and connections, and drilling down to more detailed server information.

Application Dependency Mapping Challenges

Server Consolidation and Virtualization

Companies generally prefer to use a separate server for each of their individual workloads. This helps limit the impact in the event a server goes down, needs to reboot, or is affected by a security breach.

But this comes at a cost.

With only a single application to support, a typical physical on-premises server runs at a fraction of its capacity. And this wastes valuable IT resources.

For more than a decade, the solution to the problem has been server consolidation, whereby resources are divided up into a number of fully isolated virtual machines. This allows different applications to run concurrently on the same physical hardware, each with its own independent guest operating system.

Although many organizations have long wanted to realize the benefits of server consolidation, they have been held back by the lack of visibility into their applications and their corresponding dependencies. This meant they were insufficiently equipped with the information they needed to migrate these applications to more cost-efficient virtualized environments.

The Enterprise Struggle with Server Consolidation

According to Forrester research into achieving IT efficiency through dependency mapping, 53% of the enterprises it surveyed viewed server consolidation and virtualization as a significant challenge because of a lack of visibility.

More than half of these (56%) reported they'd had an incomplete view of dependencies between applications and between applications and the underlying infrastructure.

However, ADM tools provide companies with the all-important insights to make their virtualization projects a success. They can help users fully understand the anatomy of the applications they want to move, what exactly they'd need to migrate, the interrelationships between application components, and the corresponding resources they'd need in their new environment.

Incident Management

The goodwill of customers and productivity of the IT support team depend heavily on the way it's able to deal with incidents when things go wrong.

Technical support staff should have a quick and efficient means of resolving matters, such as when:

- the company website has gone down
- an application or website is running slowly
- unexpected error messages appear on screen
- users are unable to log in or access certain services or features
- technical problems arise with online purchases

Application dependency mapping can help them do just that by giving them a way to get to the root cause of issues.

With the aid of an ADM solution, they can often see at a glance:

- what services are unresponsive
- where there are bottlenecks in system
- why a dependency issue occurred
- when it first appeared
- which team would be responsible for resolving it

In other words, ADM tools provide a wealth of information that can facilitate a quick resolution and thereby play a useful role in maintaining user satisfaction, reducing customer churn, and helping to keep support costs down.

Application Modernization and Technology Standardization

The goal of application modernization is to get more business value from IT systems by making cost efficiencies, improving performance, and taking advantage of the latest technologies, such as cloud computing, open source, and web-friendly coding languages.

Similarly, technology standardization helps companies to realize many business benefits. By adopting a common approach to application design, using standard formats, software versions, and configurations, and restricting hardware to a limited number of different manufacturers, a company can significantly reduce complexity.

This streamlines the task of managing IT as it:

- minimizes the amount of different software and hardware an organization needs to support
- reduces the range of technical knowledge needed to troubleshoot problems
- makes it easier to spot anomalies that provide tell-tale signs of issues
- improves the interoperability between different application dependencies

Application dependency mapping helps IT departments take stock of their inventory of hardware, software, and dependencies so they can accurately and comprehensively scope modernization projects and identify standardization opportunities across the organization.

A Dependency Mapping Example

The following screenshot, taken from Faddom's own ADM solution, shows an example of a dependency map for a demo ERP system hosted in a hybrid-cloud environment.

The navigation tree on the left-hand side shows two different applications, a web portal and the demo ERP system, that use a specific server. This helps the user to evaluate what impact a server event, such as scheduled maintenance or a security incident, might have on those applications.

Nodes

In our maps, nodes denoted by blue circles represent single servers while those denoted by yellow circles represent clusters. Each node has two names, where one is the domain name and the other is the virtual machine name taken from VMware.

Connections

Each arrow represents a connection between two nodes. The direction of the arrow indicates which node is the one that initiated the connection between them. In other words, whether a connection is incoming or outgoing from the perspective of each node.

Faddom uses the following color scheme to show the status of each connection:

- **Green:** An active connection that has seen traffic in the last few minutes, the exact period of which can be set by the user
- **Grey:** An inactive connection, which hasn't seen traffic over a longer predefined period
- **Red:** A failed connection, where a known application component isn't responding

Such maps help users document all incoming and outgoing connections to a server and see the current status of each connection.

ADM Best Practices

In an earlier part, we explained why application dependency mapping is important and how it can help users to adopt dependency best practices. This, in turn, will help prevent many types of dependency issues from arising in the first place.

In this section, we take a closer look at some of the best practices that development and operations teams should keep in mind.

Use OS-Agnostic Code

The less software code relies upon the underlying operating system then the more it will be able to tolerate changes to and issues with dependencies elsewhere in the application stack.

Applications should therefore avoid using services provided by the operating system wherever possible. And, likewise, they should avoid using other interfaces that are likely to change or deprecate over time

Containerize Workloads

Another way to ensure agnostic code is to deploy it to containers.

As with virtual machines, containers provide isolated environments for hosting applications but use an alternative method of virtualizing infrastructure resources.

Instead of using a hypervisor to create and run virtual machines, containers share the kernel of the host operating system with other containers. They do this by leveraging the namespaces feature of Linux to create isolated processes, each with its own set of partitioned resources.

Containers effectively decouple applications and their dependencies from their host environment. And, as a result, they tend not to be impacted by changes elsewhere in the software supply chain.

Keep Tabs on Open-Source Dependencies

The wrong choice of component can lead to potential licensing, security, and compatibility issues. Not only that but it's also important to consider the viability of some open-source technologies over the long term.

This is particularly so where open-source projects are the work of only a small community of contributors. This makes them vulnerable to open-source burnout—a situation in which contributors struggle to keep up the commitment of developing and maintaining the software until eventually the project winds down completely.

This can leave an organization with the choice of either fixing code defects and vulnerabilities themselves or switching to an alternative solution that provides the same functionality.

So it pays dividends to carefully vet each choice of open-source dependency by checking how often it's updated and for other signs of long-term sustainability.

Minimize Use of Proprietary Dependencies

Proprietary dependencies can be even more problematic. So avoid using them where practical and possible.

This is because vendors don't provide access to their source code. As a result, organizations are left with far less latitude when dependency problems arise and are often at the mercy of the vendor's customer support service to help get them resolved.

Proprietary dependencies also increase the risk of vendor lock-in, whereby customers find it increasingly difficult to switch to alternative solutions and have to pay for expensive upgrades when vendors drop support for earlier software versions as part of the product lifecycle.

Not only that but vendors may also discontinue products for commercial reasons and even potentially go out of business at any time.

Make Dependencies More Visible and Easier to Discover

To get the best results from any mapping exercise, it's important to make it as easy as possible for ADM tools to discover, understand and accurately map dependencies.

This should start with clear, comprehensive and up-to-date documentation and good use of tagging, naming conventions, and commenting in code. These measures will help improve the preliminary knowledge users are able to feed into an ADM solution so it can map an application ecosystem to a higher level of precision.

Visibility will also depend on smart management of access controls, so mapping tools can capture the information they need without compromising security.

And, in addition, development teams should bear in mind that application architectures should not only meet their technical objectives, but also make it easy for others to understand the topology of their designs.

Take Different Types of Dependencies into Account

Finally, given the multitude of different dependencies that make up the modern application supply chain, it's important to ensure nothing slips under the radar.

And this isn't just limited to technology.

For example, BCDR planning will need to consider company politics when setting priorities for restoring systems following an IT disaster or other disruptive event.

Application dependency mapping can once again help by showing stakeholders across the business how each of their applications form part of a wider picture, where one system may depend on one or more others in order to function.

This can help overcome objections from users of an application that's lower in the company's list of recovery priorities.

Faddom's Application Dependency Mapping Software

Faddom helps companies worldwide with our agentless software that maps entire hybrid IT environments — both in the cloud and on premise — in as little as one hour. Start a free trial today!

Learn More About Application Mapping

Why IT Mapping Needs an Application Dependency Mapping Tool

IT mapping is critical to keep track of infrastructure. But digital transformation is exposing the limits of many ways to do so.

Read more: Why IT Mapping Needs an Application Dependency Mapping Tool

Mapping as a Critical Active Directory Migration Tool

Even the most comprehensive AD migration tool cannot ensure the discovery of all aspects of the environment as part of the migration process.

Read more: Mapping as a Critical Active Directory Migration Tool

The Role of IT Service Mapping in the Digital Enterprise

IT needs a complete understanding of how applications and systems deliver business services. This includes a detailed view of the infrastructure.

Read more: The Role of IT Service Mapping in the Digital Enterprise

An Introduction to Network Mapping Tools & Software

All companies need well-functioning networks. Outages and bottlenecks can annoy everyone and interfere with their jobs and product use.

Read more: An Introduction to Network Mapping Tools & Software

How to Monitor Network Traffic & Get Full Network Visibility

Learn how to monitor network traffic to gain full network visibility on VMware virtual environments with our four-step illustrated guide.

Read More: How to Monitor Network Traffic & Get Full Network Visibility

Network Topology Mapping 101: The Categories, Types, and Techniques

Network topology tracks a telecommunication network's density and functionality. This post explains its categories, types, and techniques.

Read more: Network Topology Mapping 101: The Categories, Types, and Techniques


FADDOM