PRADITA University

MTI104 - IT Services

Session-01:

ITIL Introduction and DevOps Overview

PRU/SPMI/FR-BM-18/0222

Alfa Yohannis

Introduction to ITIL 4

- ITIL has evolved through multiple versions.
- ITIL 4 offers a fresh perspective on service management.
- ITIL 4 addresses the blurred line between development and operations.
- The framework adapts to the digital age.
- ITIL 4 replaces outdated practices from ITIL V3.
- It focuses on agility and innovation.
- ITIL 4 was developed to remain relevant in the fast-changing IT world.

Why ITIL 4?

- ITIL's evolution from V3 to 4 reflects industry changes.
- Initial concerns were about certification relevance.
- ITIL V3 was widely successful but became outdated.
- New ITIL 4 framework needed to address digital transformation.
- ITIL 4 aims to modernize service management practices.
- The refresh was seen as overdue but necessary.
- ITIL 4 adapts to new industry dynamics and practices.

ITIL V3 Limitations

- ITIL V3 was seen as outdated in the digital age.
- Its traditional framework lacked agility and dynamism.
- It was criticized for its process-driven approach.
- The service lifecycle model became irrelevant for modern needs.
- ITIL V3’s phases were too rigid for fast-paced environments.
- The framework was compared to outdated technologies like Nokia’s phones.
- The need for a more flexible and responsive approach led to ITIL 4.

DevOps and ITIL 4

- DevOps bridged development and operations.
- It replaced traditional blame games with collaboration.
- DevOps methodology emphasizes agility and rapid delivery.
- ITIL V3's rigid processes conflicted with DevOps practices.
- ITIL 4 integrates with DevOps by adopting agile principles.
- New frameworks needed to support continuous development and operations.
- ITIL 4 adapts to modern workflows and product management needs.

ITIL 4 Key Changes

- The service lifecycle concept is replaced by the Service Value System.
- Introduction of practices that encompass processes and tools.
- Functions from ITIL V3 are integrated into practices.
- Service definition now includes value co-creation with customers.
- Governance has a more explicit role in ITIL 4.
- Automation is emphasized for efficiency and integration.
- The certification path is updated to reflect new roles and practices.

ITIL 4 Certification Hierarchy

PRADITA University

- ITIL Foundation is the entry-level certification.
- ITIL Managing Professional (MP) focuses on service management.
- ITIL Strategic Leader (SL) looks at business strategy.
- ITIL Master certification is for advanced practitioners.
- Certifications cover various aspects of ITIL and digital strategy.
- ITIL 4 certifications align with modern service management needs.
- The hierarchy helps professionals choose relevant certifications.

ITIL Conclusion

- ITIL 4 represents a significant update from V3.
- It addresses the needs of modern IT environments.
- Emphasizes agility, collaboration, and value co-creation.
- Adapts to industry shifts towards DevOps and digital transformation.
- Provides a more flexible and relevant framework for service management.
- Updated certification paths reflect evolving professional roles.
- ITIL 4 aims to stay relevant in the fast-evolving IT landscape.

Brief Overview of DevOps

- New methodologies often emerge to solve problems.
- DevOps was created to address the need for fast solution turnarounds.
- Businesses faced issues with incomplete information during development.
- DevOps evolved to enhance agility and productivity.
- It aims to improve software quality over time.
- Operations also benefit from DevOps through increased automation.
- Automation transforms mundane tasks into innovative processes.

The Evolution of DevOps

PRADITA University

- Development was managed through waterfall project management.
- ITIL was dominant in operations before DevOps.
- DevOps merges development and operations.
- Waterfall methodology gave way to Agile practices.
- Concerns about ITIL's role in DevOps were unfounded.
- DevOps is not the end for ITIL but a complementary approach.

What Exactly Is DevOps?

PRADITA University

- Multiple definitions exist for DevOps.
- Commonly understood as integrating development and operations teams.
- Initially thought to be just about automation or startups.
- Increasingly recognized as a cultural change.
- DevOps involves more than just combining teams.

DevOps Explained with an Example

PRADITA University

- Example: Rollback of a failed deployment.
- Root cause analysis and fixing involve multiple stages.
- Blame culture vs. DevOps blameless culture.
- In DevOps, responsibility is shared among the team.
- Mistakes are viewed as part of human nature.

Why DevOps?

- Evolution from sequential waterfall to Agile methodologies.
- Agile introduced flexibility but still had limitations.
- DevOps enhances Agile with automation for faster delivery.
- Automation improves efficiency beyond what Agile alone can offer.

Benefits of Transforming into DevOps

- Enhanced working culture and technology.
- Organizations like Amazon set new benchmarks in deployment.
- Amazon statistics: 1,079 deployments/hour, 75
- Automation and reduced downtime are key benefits.

DevOps Principles

- Principles are evolving and include CALMS.
- CALMS stands for:
  - Culture
  - Automation
  - Lean
  - Measurement
  - Sharing

DevOps Scope and Elements

- DevOps is a cultural change, not a framework.
- It includes people, process, and technology.
- Implementing DevOps spans the entire software lifecycle.
- Key elements: People, Process, Technology.
- DevOps practices can be applied to various parts of the software industry.

Brief Overview of DevOps

- Conjunction of development and operations
- Change in software delivery culture
- People at the heart of transformation
- Integration of development and operations teams
- Resolving conflicts and mistrust
- Balancing development and operations priorities
- Creating communication channels

Integration of Development and Operations

- Teams amalgamated for cultural change
- Development integrates with operations
- Reduced issues in CAB
- Operational familiarity with development
- Better software maintainability
- Progressive collaboration
- Increased trust and efficiency

Conflict between Development & Operations

- Development team vs. operations team
- Cliff-like separation between teams
- Unpredictable activities in between
- Miscommunication and mistrust
- Need for a bridge between teams
- Creation of shared responsibility

Balancing Priorities

- Development focus: new features
- Operations focus: stability
- Development introduces change
- Operations ensure environment stability
- Continuous changes impact stability
- Balance between innovation and stability
- Continuous improvements and maintenance

Creating Channels of Communication

RADITA
niversity

- Merging development and operations
- Shared responsibility for smooth operations
- Communication within team members
- Joint work on development and deployment
- Reduction of mistrust
- Improved deployment confidence
- Enhanced collaboration

Process

- Key to project success
- Focus on processes over automation
- Automation should follow processes
- Define processes first
- Align with DevOps architecture
- Tools and automation support processes
- Avoid process-driven tool adoption

Adapting Processes for DevOps

- Adjust processes for new objectives
- Transition from traditional methodologies
- Agile methodologies preferred
- Flexibility and adaptation in project management
- Importance of process in DevOps
- Agile principles and practices
- Rejigging processes for efficiency

Technology

- Technology as a key DevOps element
- Importance of automation
- People and processes must come first
- Technology supports efficiency
- Proper synchrony of roles and processes
- Tools enhance DevOps practices
- Avoid technology-driven development

DevOps Practices

PRADITA University

- Continuous Integration
- Continuous Delivery
- Infrastructure as Code
- Continuous Monitoring
- Integration and deployment practices
- Automation in software management
- Real-time monitoring and feedback

Continuous Integration

- Regular code integration
- Detect conflicts early
- Minimize rework
- Frequent integration checks
- Automated builds and tests
- Resolve conflicts in real-time
- Maintain code quality

Continuous Delivery

- Automated deployment to production
- Maintain deployable state
- Reduce deployment effort
- Frequent and reliable releases
- Rapid feedback and fixes
- Minimize deployment risks
- Improve feature delivery speed

Infrastructure as Code

- Automate infrastructure management
- Define infrastructure with code
- Version control for infrastructure
- Consistency and repeatability
- Reduce manual interventions
- Manage resources through configuration
- Enable scalable infrastructure setup

Continuous Monitoring

- Ongoing performance analysis
- Real-time metrics collection
- Identify issues promptly
- Maintain system stability
- Proactive issue resolution
- Monitor application health
- Improve system performance

DevOps Conclusion

- DevOps integrates development and operations to improve agility.
- Evolved from traditional methodologies like Waterfall and ITIL.
- Emphasizes cultural change and shared responsibility.
- Enhances Agile practices with automation for faster delivery.
- Benefits include increased efficiency, reduced downtime, and improved software quality.
- Key principles: Culture, Automation, Lean, Measurement, Sharing (CALMS).
- Effective DevOps involves people, processes, and technology across the software lifecycle.