

View Point



Building a Robust Suite for Automated Regression Testing of SAP Releases

Surendra Dosad, Nishanth Rao & Ravi Rengasamy

If your organization needs to implement automated testing, the right approach and strategy can help cut time and cost on SAP releases

Abstract

The world's most popular ERP package, SAP R/3, offers enormous flexibility in configuring applications. This advantage however comes with risks attached in terms of the impact releases can have on business owing to the sheer size, complexity and demands for shorter implementation cycles, besides obvious financial implications. This raises the bar for effective and extensive quality assurance (QA) of key business processes and configuration in ever shortening cycles before a release can 'go live'.

If your organization truly needs automated testing then timely QA testing of SAP applications along with legacy systems will ensure that all systems work together reliably. The key to building a robust solution for automated regression testing is selecting the right test tool and automation framework, besides choosing the right business scenarios and test cases for testing.

Is Automated Regression Testing Needed?

In this dynamic, hyper-competitive information age, quick implementation of key business applications can help reduce costs and increase market share. The need to meet ever aggressive delivery schedules exerts increasing pressure on implementation, as does the demand to reduce costs. While reaching the market late can prove expensive in terms of competitive advantage, delivering defective software products could be catastrophic. If your organization faces similar challenges, test automation can help reduce time to market while cutting costs, providing a practical way to achieving quality objectives within the constraints of schedules and budgets.

Although companies are increasingly adopting the test automation route to quicker releases, the downside can be huge without the right strategy and approach. A roadmap for successfully building a robust regression testing suite for SAP releases can reduce cost while providing reliable QA.

Challenges Organizations Face in Typical SAP Releases

- Time constraints for development and QA cycle
- Large number of test cases/ scenarios for execution to ensure sufficient and effective QA for critical business processes
- Large number of changes (fixes/enhancements) with each release of an application
- Distributed and diverse legacy applications
- Risk associated with “Ripple effect” of configuration changes across R/3 application modules
- Manual testing requires large number of testing staff, results in higher number of errors, provides less test coverage, offers inconsistent results and is time-consuming

An organization dealing with the above challenges can benefit greatly from automated regression testing if it is implemented properly using the right strategy, processes and approach.

How Automation can Address these Challenges

- Reduced cost of failure by early detection of bugs and issues during testing
- Reduced cost, by completing the QA cycle quicker
- Automated testing is consistent and repeatable
- Automation routines can be executed for data setups like creation of master data & variants
- Reporting and critical defect detection inputs for impact analysis
- Automated tests can extract variable data from one application and pass it to a test to run in another module, ensuring different application modules and interfaces work well together
- Once created and benchmarked, the automation suite needs minimum manpower to execute tests during each cycle, eliminating the need for experts in all SAP modules
- Automation is the way to cost-effectively and consistently achieve adequate test coverage. It provides proof of compliance to law with strict quality regulations and audit trail standards
- Automation ensures reliability of mission-critical business processes like financial month-end closings, bank reconciliation, production planning, sales order entry, etc
- As new versions of R/3 are implemented and “hot-packs” applied, automated tests can be reexecuted for regression testing for critical business processes to be performed without error
- Test suite can be used for regression testing while legacy system software, networks, server hardware, and communication components are upgraded to prevent a negative impact
- Automation ensures uniformity in the testing process each time a test is executed
- Scripts can be executed at night or on weekends unattended – huge value in time crunch
- Test scripts can run independently with minimal human intervention in multiple sessions

Given the challenges, it is clear that automated regression testing is not a luxury but a necessity. Having identified these challenges and decided that automation is the way to go, it becomes imperative to develop the correct roadmap. The first step in this path is choosing the right test tool and automation framework.

Choosing the Right Test Tool & Automation Framework

Success in automation hinges on the right test tool & automation framework for the application.

Choosing the Right Tool

To ensure end-to-end testing of all business transactions, the selected test tool should be able to automate testing in systems other than SAP, since there are usually several legacy systems running along with SAP for various business transactions. If required, more than one test tool can be used to automate testing across systems.

Parameter	Mercury Suite Quick Test Pro (QTP 8.0)	CATT
Ease of use	GUI-based	GUI-based
Process of capturing test cases	Records SAP transactions online (front end)	Records SAP transactions online (front end)
Data types	Facility to define and change static to dynamic and vice versa	Facility to define and change static to dynamic and vice versa
SAP advantage	Official third party application	Part and parcel of SAP
Web testing support	Provides cross browser testing	Limited by SAP front-end features
Third party application testing	Facilitates third party application testing	Not possible
Testing methodology	Facilitates object-based testing independent of field positions in screen layouts	Screenshot-based testing
1. Test analysis 2. Test reporting 3. Defect tracking	Requires additional module - Test Director - which Integrates well with QTP	Detail logs available for analysis but cannot manage testing beyond SAP
Support & ease of implementing the automation framework	QTP supports all the popular frameworks	Implementing automation framework using CATT will not be easy
Technical & Licensing Information		
Scripting language	SQA Basic, a language like VB	ABAP
Cost & Licensing	> Pay per use <ul style="list-style-type: none"> • Engagement-based • Perpetual licensing 	Part of standard SAP, hence no additional cost

Support and ease of implementing the selected automation framework with the tool is another criterion for tool selection. After detailed consideration and evaluation of popular test tools from different vendors, the Mercury Suite and CATT tools have been identified as best suited for SAP automated testing.

Quick Test Pro (QTP) provides good support and ease of testing with SAP as well as third party applications, making it a popular commercial tool. Besides, there is excellent technical support from the vendor as well as the wider user community. Hence, QTP is a better solution over CATT.

Picking the Right Framework

The three automation frameworks considered for SAP automation include Functional Decomposition, Keyword Driven, and Hybrid:

- **Functional Decomposition:** Application-under-test split into different functions; reusable library of functions/ components representing small sections/ modules of applications, common routines, error-handling code. Functions called from test scripts as needed.
 - Strengths: Quick & easy to implement, popular, supported by QTP and other tools.
 - Weakness: Not as robust as Keyword Driven framework
- **Keyword Driven:** Requires the development of data tables and keywords, independent of test automation tool and development of Master test script code and keyword functions that “drive” the application-under-test and the data.
 - Strengths: Robust framework, can be deployed independent of test tool and application under test
 - Weakness: Detailed steps for each test condition need to be documented, not as easy as Functional Decomposition; more initial time required
- **Hybrid:** Over time, most frameworks evolve into the hybrid framework – a combination of the above two frameworks, which brings together their strengths and mitigates their weaknesses. In terms of speed and ease of implementation, the Functional Decomposition framework is ideal for SAP with QTP 8.0 as the test tool.

Identifying the Best Test Management Tool

Ease of integration of the functional testing tool is a key criterion for selecting the Test Management tool required to manage test scripts, results, and defects. Infosys recommends Test Director (Quality Center) from the same vendor, i.e., Mercury Interactive, whose products are well suited for automation of regression testing of SAP releases.

With the right testing tool and framework in place, it is important to select the right business scenarios for automation.

Selecting Business Scenarios/Test Cases for Regression Testing

The key to success in automation at this stage is identifying what needs to be automated. This calls for a clear definition of the goal of automation in terms of which scenarios or business processes be considered for automation and the purpose of the automated regression test.

Automation Goal

SAP releases are repetitive. To test all business transactions, interfaces, configurations, etc., is not possible in every regression test cycle, which typically lasts for less than a week.

The following factors are critical in determining the goal of automation:

- Sufficient & efficient QA to ensure critical path of business always up and running
- All key interfaces spanning various systems and SAP always up and running
- Fixed budget & time constraint for regression testing during each testing cycle

Based on the available time for QA during each regression cycle it is recommended not to automate all business transactions/ scenarios. Rather, the focus of regression testing is to ensure the critical path of the business is not impacted due to new code moving in production.

Take a car that has been modified and is to be released in the market. The four most critical operations would be: (a) Start engine (b) Steer the vehicle (c) Provide acceleration (d) Apply brakes. Testing these functions is critical. Time and budget permitting, additional tests can be performed.

Identifying Business Scenarios /Test Conditions

The goal for regression testing is to test and ensure the critical path of business is not impacted. To identify and define the scenarios, which fall under the critical path, joint meetings are essential between the Business, Process, Project Management and QA Team.

All business scenarios for different process areas must be identified. Each scenario should be discussed & evaluated, identifying the most critical businesses scenarios which should never be impacted.

These business scenarios spanning different SAP modules/ systems form the requirements for building the automated regression testing suite.

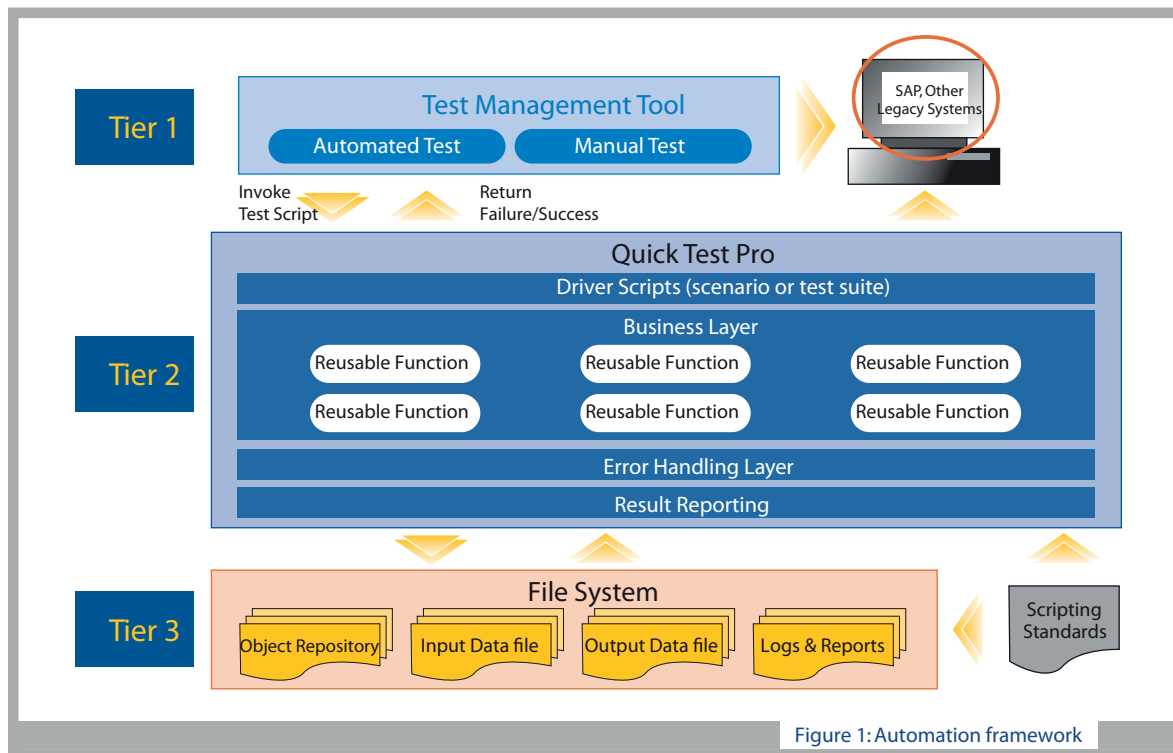
The following factors should also be considered for selecting the right test cases/scenarios for test automation:

- Is automation feasible for the selected scenarios/ test cases?
- Identify areas of ERP that have been customized as customizations can have an adverse effect on functionality and performance that must be tested with every build
- Concentrate testing on interfaces with communication between SAP & legacy system
- Scenarios/transactions which need to be tested with different data sets/values
- High path frequency: Scenarios/process paths used with high degree of frequency when the software is running in full production. Examples include creating customer records, invoicing and other high volume activities where software failures can occur frequently.

Having chosen the right business scenarios for testing, the next step is designing the appropriate automation architecture to build the test suite.

Building the Test Suite: Automation Architecture & Key Considerations

The figure depicts an automation framework and its components:



Tier 1: Tier 1 comprises elements like Test Management Tool (Test Director) and System-under-test, i.e., SAP and other legacy systems.

Tier 2: Tier 2 comprises test tool QTP and its elements like Driver Script, which in turn calls reusable function with business logic layer. It has one more layer for error handling and reporting logic.

The main components of Tier 2 are:

- **Error Handling Functions:** Error handling functions are the basic component which take care of unexpected errors, warning or information messages during execution of scenarios in SAP.
- **Reusable Function:** Reusable function refers to a unique SAP transaction code (T-Code) in SAP. All T-codes are scripted as functions that form part of the reusable library.
- **Driver Script:** The driver script is the one that actually runs a test condition in a particular scenario. In turn, it calls different reusable functions from the reusable library.

Tier 3: Tier 3 consists of the file system and scripting standards.

- **File System:** File system components include:
 - **Object Repository:** This repository in QTP includes physical description of objects present on SAP application GUI. Convention: Object repository is per reusable action
 - **Input Data File and Output Data File:** All required test data separated from test scripts. Required test data is stored in different xls. Master data file consists of data common across scenarios. Test data for each business scenario stored in one xls.
 - **Logs and Reports:** This framework component stores results and logs errors if any during execution of script. Success/error messages logged into one result log xls.
- **Scripting Standards:** All the scripts should follow scripting guidelines and coding standards defined during development of reusable functions and driver scripts.

Key Considerations for Developing Automation Scripts Architecture for SAP

- To automate all critical business end-to-end scenarios, each scenario is split into test conditions, and each test condition broken down into unique SAP T-Codes that must be executed
- Each unique SAP transaction, i.e., T-Code, is scripted as a reusable action under the Common Reusable library.
- Each reusable action is called in different scenarios with the help of a driver script. All the required parameters to execute the reusable action (T-Code) are passed through the driver script. Error handling, common functions also built into reusable common library; included in each script
- Using test management tool, test sets are created for each business scenario; test conditions are added to test set to complete testing & validation of one end-to-end business process
- All scenarios required to be tested in each regression cycle are set for execution using individual test sets driven by test management tool
- The structure of having of one test script per test condition provides great modularity and ease of maintenance and debugging. If a test condition fails, it is easy to detect which test condition and corresponding test script in the complete business scenario caused the issue and broke the execution
- Separate the test data from the scripts. Maintain all the required Master & Transactional data in separate files. These files input the required data to the scripts during execution
- Estimation & planning of effort required to completely build the regression test suite becomes easy. Calculate the effort required to build script for each unique T-Code, common reusable library functions, error handling functions, developing of calling scripts and integration of individual test conditions into scenarios
- Proper coding & commenting guidelines should be documented; all team members must follow the same
- Peer review process to ensure all scripts are well reviewed
- Ensure all the required test data (Master and Transactional data) are current and valid.
- Perform dry run to integrate and run scripts to test a complete business transaction
- Proper documentation for each test script

- Apply configuration management process and tool to ensure test scripts and artifacts are version controlled and well maintained
- Treat automation project like any other development project

Once the automation architecture has been finalized, the next stage is the execution and validation cycle.

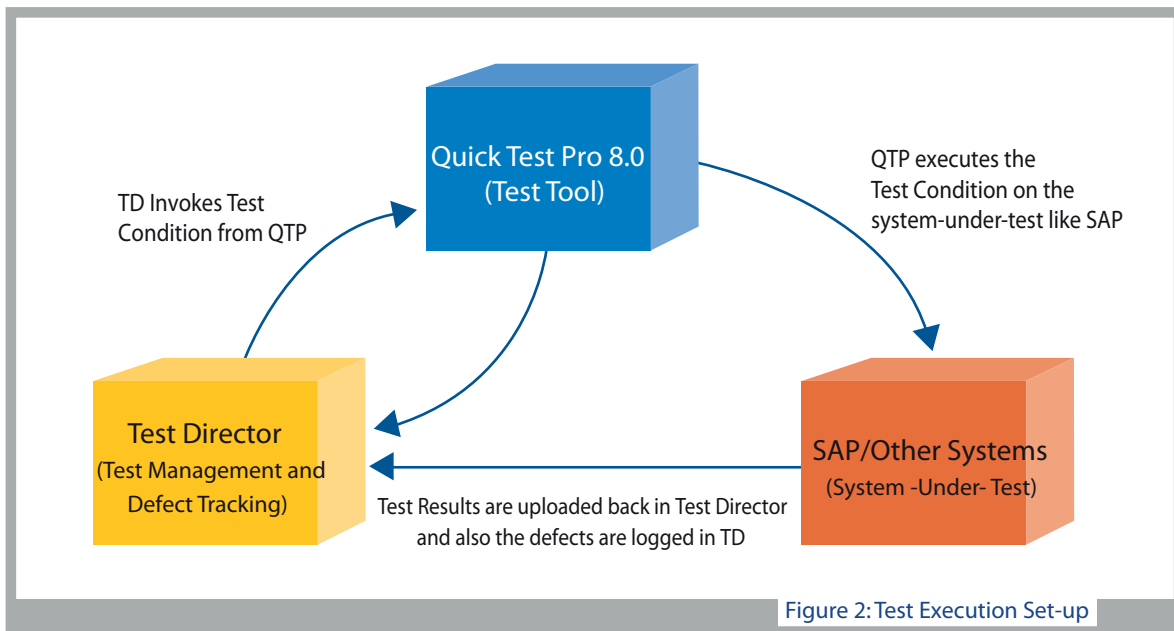
Execution and Validation Cycle of Test Suite

Understanding Business Scenario before Building Automation Script:

One of the challenges test automation engineers face in complex SAP testing is understanding the complete business process and test cases. This is the key to building effective and robust test scripts for validating each critical business transaction. This can become time-consuming. To speed up this process:

- During each regression cycle, testers should execute complete scenarios with the help of Business/SMEs manually. This will help the tester understand each test condition and the end-to-end business scenario completely and effectively.
- At the time of execution, testers should update test cases in case of changes and also make notes that will help build the automation script for the scenario.
- Before the next cycle, all scenarios executed/ tested manually during last regression cycle should be scripted and robust automation script built. During next cycle, the automated scenarios are tested with automation script and new scenarios executed manually with Business/SME. Process to continue until all critical business scenarios are automated and complete automation suite built.

Test Execution Set-up:



The test execution set-up as shown in Figure 2 has three main components:

- **Test Director (TD):** This is an automated script management tool. This tool also provides defect tracking, test set execution and result reporting.
 - During each regression cycle, test set created in TD and all scenarios and test scripts pertaining to the scenario added in test set.
 - All the test scripts within each scenario are executed in batch mode from test set in TD.
- **Quick Test Pro:** QTP is the test automation tool. Whenever test condition is executed from TD, it invokes QTP and loads test script in QTP which executes script on system-under-test (SAP/other system).
 - Enables settings in QTP to capture and upload test results in TD.

- This ensures required screen shots and test results are saved in TD for later verification and validation.
- **System-under-test:** SAP or other systems with which SAP interacts to complete business transaction. QTP executes test scripts on system-under- test.

Smoke Test:

- Smoke test is a popular testing practice executed before actual regression testing.
- It comprises automated test scripts of key SAP transactions used in scenarios.
- Before actual regression testing can begin, , conduct “Smoke Test”. This ensures all required systems are available and up & running.
- Thus a smoke test helps detect key SAP transactions and environment related issues earlier rather than during execution of a critical end-to-end business transaction, saving significant time and effort in executing test scenarios during actual regression

Result Reporting and Business Sign-off using Test Director:

- Before regression testing, test execution and validation schedule is published for key stakeholders. Required details like scenarios under execution (automated or manual), date of execution, names of tester and SME responsible for validating and business sign-off for each scenario are published well in advance.
- On execution of scenarios, results and screen shots are automatically saved in Test Director (TD) by QTP and presented by testers to business/SME for business sign-off.
- SME can log in TD and view complete test execution results to provide feedback and recommend corrections in test condition/scenario or test process. If there are no issues he can provide business sign-off.
- This ensures user acceptance testing by business, adding review layer to the testing process before the SAP release goes into production, and better co-operation and understanding between the SAP process, business and QA team.
- Saved test results/screen shots are useful for future reference and process audits.

Maintenance of Test Suite

Once the automated scripts are developed, regular maintenance is needed to keep scripts updated and running. Defining a process to maintain the suite is imperative if ROI is to be maximized. The Infosys process to keep scripts & test suite current is given below.

Inputs for Changes/Updates to Script

Any automation script may need to be updated due to:

- Change in business process/logic
- GUI modification in SAP screens
- SME feedback from test execution
- New transports and enhancements in application
- Changes made in external systems integrated with SAP
- Errors in the scripts

Change Process

- All inputs for modification to scripts are documented and reviewed, changes prioritized and effort estimation & impact analysis carried out
- It is important to locate the exact lines in scripts for updates, without impacting other scripts.
- This is possible only with the right framework and architecture for the regression test suite.
- How to ensure a framework for minimal maintenance:
 - Only one script per SAP transaction; all transactions form common reusable library.
 - Scripts for each T-Code are generic; can be called in any business transaction.

- Required parameters of called T-Code reusable action pass through calling scripts. This ensures any correction in GUI is related to particular transaction code/ screen.
- Input data handled in separate data sheet for each business scenario and thus separated from the test scripts.
- Modular framework enables adding new scripts and scenarios without disturbing the existing suite; integration is easy
- Changes to test scripts under strict version control process.
- All changes reviewed and documented

Script Health Checkup

- Beside maintenance, scripts are executed periodically in batches overnight and test results analyzed. In case of failures, scripts are updated to correct errors.
- Scripts evaluated and reviewed against checklist of following parameters periodically to ensure regression suite is up to date and ready during regression testing of the SAP release:
 - Test data for each test script valid and current
 - Test environments (SAP, other systems, databases & interfaces) up and running
 - All required changes suggested incorporated and script up to date
 - Test environment. setup, connectivity, tool license, configuration and tools available
 - Test set and test scripts pulled in Test Director before execution starts
 - Test cases are up to date and scripts reflect current steps as per test case
 - Required results & defects reporting templates in place
 - Access to SAP system & transactions and other systems/databases available
 - 'All clear' in test results report and no issues identified with scripts during dry run

Lessons Learnt

Challenges in SAP Test Automation:

- Test data: Finding and building correct test data difficult without support of process and business teams
- Difficult to find suitable test tool which can automate all systems including SAP
- Automation of report validation of various reports generated in SAP not easy:
 - Contents of the report are dynamic
 - Test tools capture portion of the report displayed on the screen
- Frequent Data sync-up issues crop up between SAP and other legacy systems
- Understanding all critical business flows by one test engineer is impossible
 - All SAP transactions and processes not easily mastered by test engineer
 - Test engineer should understand other systems with which SAP interacts
- For effective regression testing, QA environment should be refreshed periodically
- Same business transaction may have different flow based on location and culture
- Many SAP transactions financial period dependent; testing stalled during closures
- While testing, some systems or interfaces may be down. Availability of all required systems may be impacted due to refresh, upgrade cycle, maintenance or QA issues
- Tester may not have access to all systems and transactions in SAP
- Not possible to execute all scenarios and test cases in each regression cycle

Conclusion

Automated regression testing is imperative for SAP releases with limited time and budget resources. By starting with the decision to go ahead with automation, challenges can be overcome with significant tangible benefits. Selecting the right set of test tools and framework, identifying the right goal for automated regression testing, using the right considerations in building the architecture, well-planned execution & validation cycles, and regular maintenance, will make automation successful for quality and timely SAP releases.

Case Study

A large automotive tier-1 supplier with 250 plants across the world was implementing SAP for its Europe and North America operations. There were multiple bottlenecks with challenges including lack of coordination between business and development teams with the testing team. Infosys defined the testing and test automation strategy in accordance with the business, created the test automation script, and executed functional and regression testing. Automated testing reduced total cycle time by 75-80 per cent, standard testing procedures resulted in better scalability, and proper documentation saved time and effort on training.

About the Authors

Surendra Dosad is a Test Manager with Infosys' Independent Validation Solutions group. His experience spans multiple testing and test automation projects across complex enterprise applications and packages like SAP in domains like finance, telecom and manufacturing, for Fortune 1000 customers. He is currently leading a QA Team testing SAP releases for a Fortune 100 automotive giant.

Surendra can be reached at surendra_dosad@infosys.com

Nishanth Rao is a Test Analyst with Infosys' Independent Validation Solutions group. His experience spans multiple testing and test automation projects across complex enterprise applications and packages like SAP in domains like retail and manufacturing, for Fortune 1000 customers. He is currently part of a QA team testing SAP releases for a Fortune 100 automotive giant.

Nishanth can be reached at nishanth_rao@infosys.com

Ravi Rengasamy is a Test Analyst with Infosys' Independent Validation Solutions group. His experience spans multiple testing and test automation projects across complex enterprise applications and packages like SAP in domains like retail and manufacturing, for Fortune 1000 customers. He is currently part of a QA team testing SAP releases for a Fortune 100 automotive giant.

Ravi can be reached at ravi_rengasamy@infosys.com

To know more about Infosys' SAP Expertise, please visit www.infosys.com/sap



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.