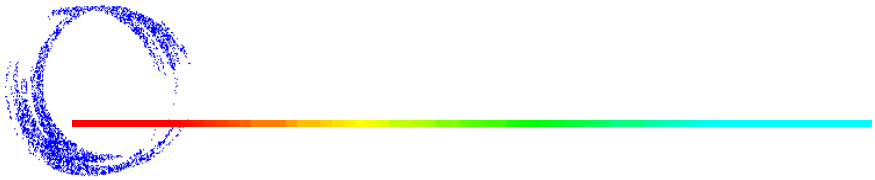


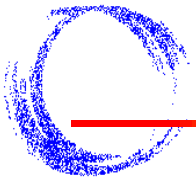
Track 7: Performance
Session 710

Managing the Performance of Composer Applications

Terry Durkin
Programart Corporation

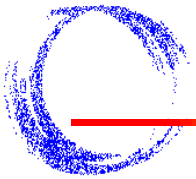
Agenda

- Application Performance Management (*APM*)
  - What is *APM*?
  - Benefits
  - *APM* levels
- *APM* for Composer/IEF
  - Environment
  - Requirements
  - Benefits
- Performance study
- *APM* Summary



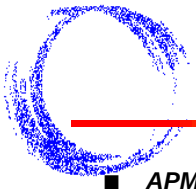
What is *APM*?

Application Performance Management (*APM*) is a discipline that allows IS organizations to deliver efficient, responsive applications and maintain high standards of application performance throughout the life-cycle.



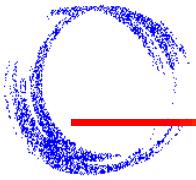
What is *APM*?

- **Two principles central to *APM*'s focus on quality**
  - **IS management must view application performance as a measure of product quality**
  - **IS functional groups must share accountability for application performance**



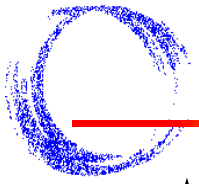
What is *APM*?

- ***APM*** **program**
  - **collection of processes an organization defines and assigns to IS functional groups**
- ***APM*** **activities**
  - **gather data on application performance in all phases of the application life-cycle**
  - **identify opportunities for improving performance**
  - **assess the impact of design decisions and coding changes**
  - **establish and maintain performance standards**
  - **track information to quantify *APM* related savings**
  - **communicate performance knowledge throughout the IS organization**

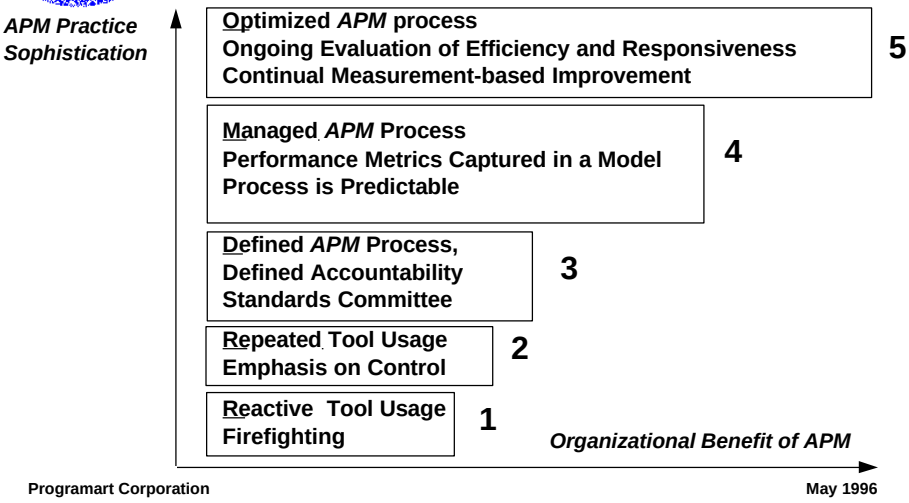


APM Benefits

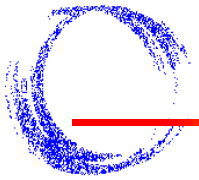
- **Ensure applications deliver maximum service at minimum cost**
- **Reclaim computing capacity**
- **Minimize the performance impact of changes in workload, technology, and business requirements**
- **Pre-empt performance crises**



APM Levels



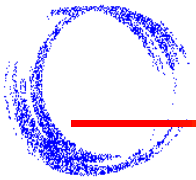
Programart Corporation May 1996



APM Level 1 - Reactive

- **OBJECTIVES:**
  - **To resolve production crises or avoid near-crises**
  - **To restore performance of production applications to promised levels**
- **IDENTIFYING CHARACTERISTICS:**
  - ***APM* tools are used on an ad hoc basis and only in reaction to existing or impending production crises**
  - **Accountability for application performance rests with production-support groups**

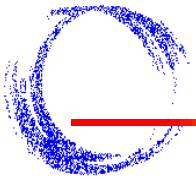
Programart Corporation May 1996



APM Level 1 - Reactive

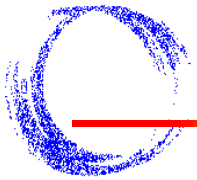
- **BENEFITS**
  - **Resolution of crises or the avoidance of near-crisis situations**
  - **Reduce resource consumption of production applications**

Level 1 is appropriate for organizations faced with existing or impending performance crises in the production environment.



APM Level 2 - Repeated

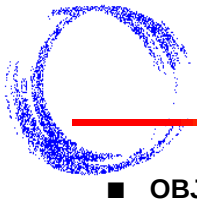
- **OBJECTIVES:**
  - **To improve the performance of production applications**
  - **To reclaim computing resources**
  - **To reduce application execution costs on an ongoing basis**
- **IDENTIFYING CHARACTERISTICS:**
  - **Targeted production applications are evaluated through systematic, repeatable projects designed to achieve specific objectives**
  - **Accountability for application performance still rests with production-support groups**



APM Level 2 - Repeated

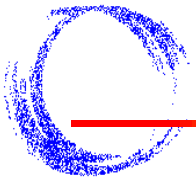
- **BENEFITS**
  - **Reduce resource consumption of targeted production applications**
  - **Less frequent occurrences of production-level performance crises**

Level 2 is appropriate for organizations that desire to improve the performance of production applications in a controlled and systematic manner.



APM Level 3 - Defined

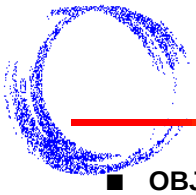
- **OBJECTIVES:**
  - **To minimize the number and severity of inefficiencies introduced into production**
  - **To maintain production applications at acceptable performance levels**
- **IDENTIFYING CHARACTERISTICS:**
  - **Both pre-emptive and reactive elements**
  - **Checkpoints mandate the evaluation of performance within the development life-cycle, preventing inefficiencies**
  - **Production applications are evaluated regularly to maintain the efficiency of critical processes**



APM Level 3 - Defined

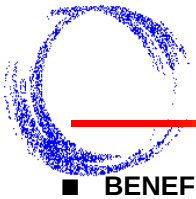
- **BENEFITS**
  - **Reduced resource consumption of targeted production applications**
  - **Reduced lifetime execution costs of applications**

Level 3 is appropriate for organizations that desire to manage application performance through structured processes.



APM Level 4 - Managed

- **OBJECTIVES:**
  - **To build and maintain optimal levels of application performance through a managed process throughout the application life-cycle**
- **IDENTIFYING CHARACTERISTICS:**
  - **Delivering and maintaining performance-oriented applications is part of a developer's job**
  - ***APM* tools are viewed by developers as critical to do their jobs**
  - ***APM* activities are managed like other business-critical programs--by tracking status, making resource-allocation decisions, and by quantifying and reporting ROI**



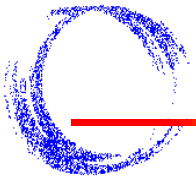
APM Level 4 - Managed

- **BENEFITS**

  By employing *APM* methods throughout the life-cycle, *APM* participants can

  - **build in efficiency at a lower application cost**
  - **sustain an *APM* program and provide regular management reports**
  - **make educated decisions about where to concentrate *APM* efforts to maximize benefits**

  Level 4 is appropriate for organizations whose management incorporates the accountability for application performance into the jobs of the IS members. It is also useful for organizations whose management regularly tracks and reports on the status of critical programs.



APM Level 5 - Optimized

- **OBJECTIVES:**
  - **To maximize *APM* benefit and reduce *APM* program costs by continually evaluating and improving the *APM* program**
- **IDENTIFYING CHARACTERISTICS:**
  - **To achieve Level 5, organizations must already have a Level 4 *APM* program in place**

APM Levels

Analysis and Design

Development

Unit Testing

System Testing

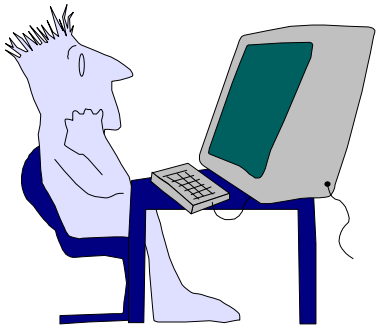
Quality Control

Production

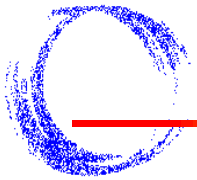
Programart Corporation May 1996

APM for Composer/IEF

How does *APM* relate to the Composer/IEF environment?

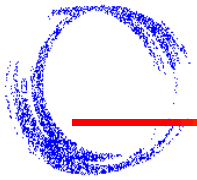


Programart Corporation May 1996

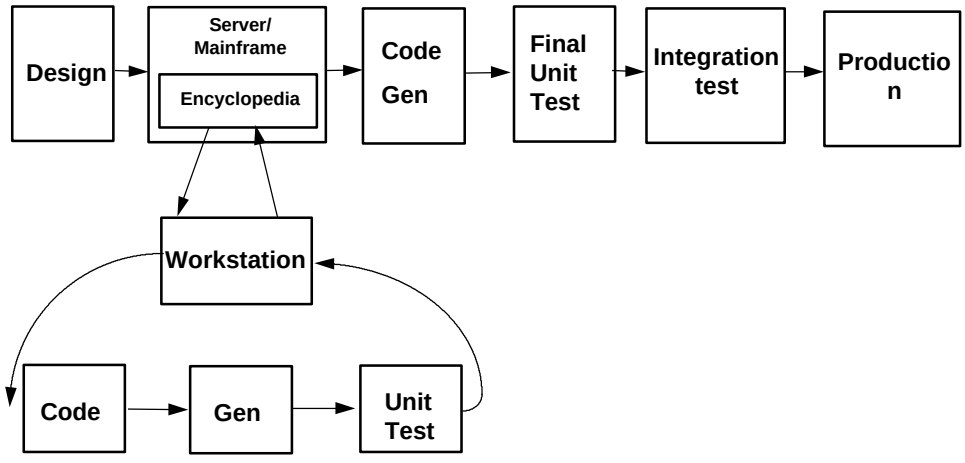


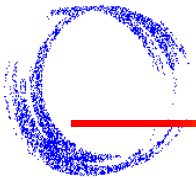
Composer/IEF Environment

- **Performance Challenges**
  - **Information Engineering (IE) Methodology**
  - **Fallacy that application performance is not a concern**
  - **Distance between developed code and executed code**



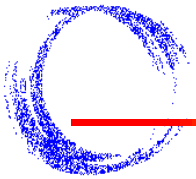
Composer/IEF Environment





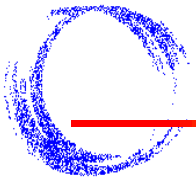
APM for Composer/IEF Requirements

- Company recognition that performance is important
- Defined goals and policies for desired performance levels
- Tools or manual procedures to:
  - Gather data
  - Identify opportunities
  - Assess impact of changes
  - Establish and maintain performance standards
  - Track information
  - Share knowledge throughout IS organization



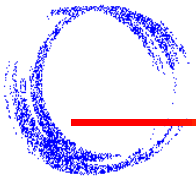
APM for Composer/IEF Benefits

- Empowers developers to take responsibility for application performance
- Helps build knowledge base for future application development
- Reduces lifetime costs of applications
- Quantifies performance improvements for management reporting



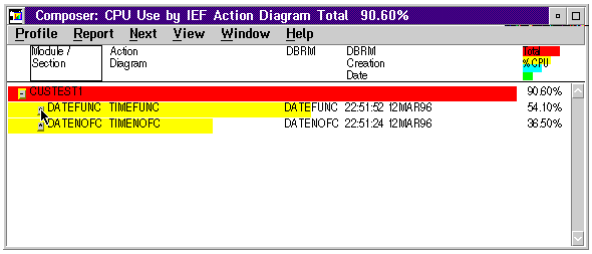
Performance Study

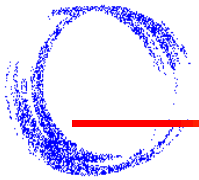
- Evaluation of Composer's time function with an *APM* tool
  - TSO application that loops through two DB2 customer tables updating a field in each record with a time stamp
  - Application contains two action diagrams
    - TIMEFUNC calls the time function in the loop for each record
    - TIMENOFC sets a work variable to the time function outside the loop and calls the work variable for each record. The work variable was defined NOT to initialize at each invocation.



Performance Study

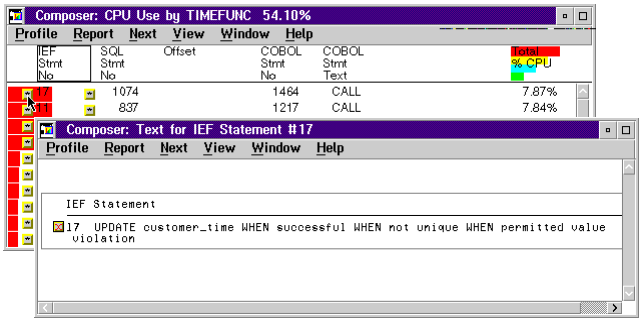
- Summary of CPU usage for all action diagrams measured

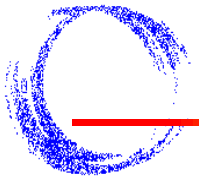




Performance Study

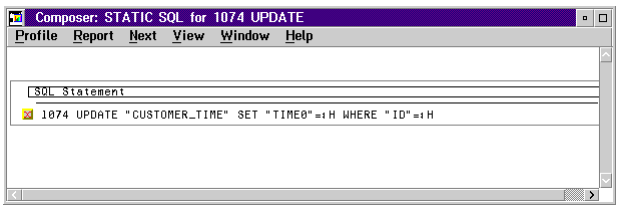
- **Detail of CPU usage by TIMEFUNC action diagram statements**

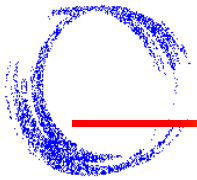




Performance Study

- **Statement text for SQL code generated by IEF statement #17**

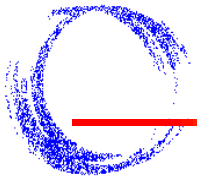




Performance Study

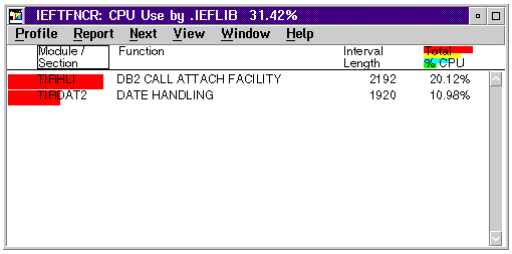
- **Listing of TIMEFUNC action diagram**
  (Caveat: listing does not follow methodology)

```
 7 |  += WHILE ief_supplied count IS LESS THAN 1000
 8 |  |  SET ief_supplied count TO
 8 |  |            ief_supplied count + 1


16 |  |  |  +- READ customer_time
16 |  |  |  |       WHERE DESIRED customer_time
16 |  |  |  |             id IS EQUAL TO "1111111111"
16 |  |  |  +- WHEN successful
17 |  |  |  |  +- UPDATE customer_time
18 |  |  |  |  |  SET time TO timetimestamp(
18 |  |  |  |  |             CURRENT_TIMESTAMP)
17 |  |  |  |  +- WHEN successful
19 |  |  |  |  |  EXIT STATE IS update_complete
17 |  |  |  |  +- WHEN not unique
20 |  |  |  |  |  EXIT STATE IS error_in_application
17 |  |  |  |  +- WHEN permitted value violation
21 |  |  |  |  |  EXIT STATE IS error_in_application
```

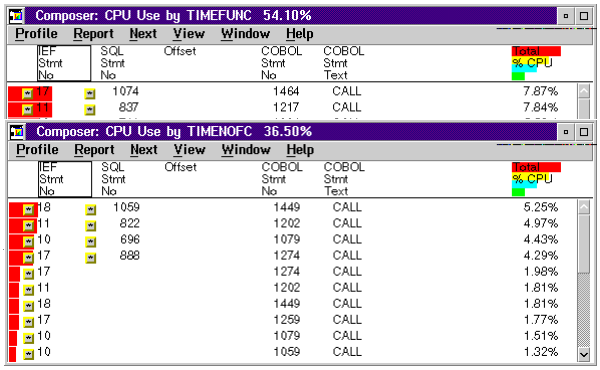
Performance Study

- **Detail of Composer services called by TIMEFUNC**



Performance Study

- **Comparison of CPU usage by statement for TIMEFUNC and TIMENOFC**



Composer: CPU Use by TIMEFUNC 54.10%

Profile Report Next View Window Help

IEF Stmt No	SQL Stmt No	Offset	COBOL Stmt No	COBOL Stmt Text	Total % CPU
17	1074		1464	CALL	7.87%
11	837		1217	CALL	7.84%

Composer: CPU Use by TIMENOFC 36.50%

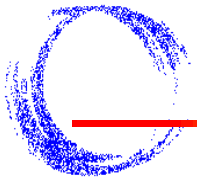
Profile Report Next View Window Help

IEF Stmt No	SQL Stmt No	Offset	COBOL Stmt No	COBOL Stmt Text	Total % CPU
18	1059		1449	CALL	5.25%
11	822		1202	CALL	4.97%
10	696		1079	CALL	4.43%
17	888		1274	CALL	4.29%
17			1274	CALL	1.98%
11			1202	CALL	1.81%
18			1449	CALL	1.81%
17			1259	CALL	1.77%
10			1079	CALL	1.51%
10			1059	CALL	1.32%

Performance Study

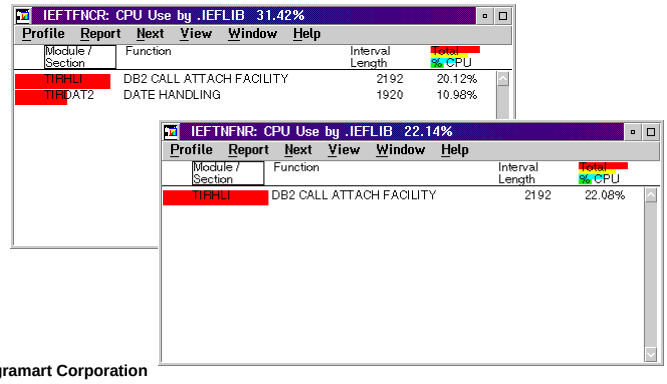
- **Listing of TIMENOFC action diagram**
  (Caveat: listing does not follow methodology)

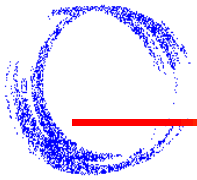
```
 4 | SET ief_supplied time_value TO
 4 |         timetimestamp(CURRENT_TIMESTAMP)
 5 | SET ief_supplied count TO 1
 6 |
 7 | += WHILE ief_supplied count IS LESS THAN
 7 | |         1000
 8 | | SET ief_supplied count TO
 8 | |         ief_supplied count + 1

17 | | | +- READ customer_time
17 | | | |     WHERE DESIRED customer_time
17 | | | |         id IS EQUAL TO "1111111111"
17 | | | +- WHEN successful
18 | | | | +- UPDATE customer_time
19 | | | | | SET time TO ief_supplied
19 | | | | |         time_value
18 | | | | +- WHEN successful
20 | | | | | EXIT STATE IS update_complete
18 | | | | +- WHEN not unique
21 | | | | | EXIT STATE IS error_in_application
18 | | | | +- WHEN permitted value violation
22 | | | | | EXIT STATE IS error_in_application
```



Performance Study

- **Comparison of Composer services called by TIMEFUNC and TIMENOFC action diagrams**





Performance Study

- **Results:**
  - **Use time function prudently; it is expensive**
  - **Determine if work variables really need to be initialized at every invocation**

APM Summary

- Application performance improvement is an ongoing process
- Application performance is the responsibility of all IS members
- Company's competitive edge and profitability are enhanced with high quality applications that are efficient and responsive

APM Summary