Q/P MANAGEMENT
GROUP, INC.

Accounting for Reuse through Function Point Analysis

Roger Heller
Q/P Management Group, Inc.
10 Bow Street
Stoneham, MA 02180
Tel: (781) 438-2692
http://www.qpmg.com
email: rheller@qpmg.com

Q/P MANAGEMENT GROUP, INC.

Accounting for Reuse through Function Point Analysis

The methodologies reflected in the enclosed material, including the benchmark comparisons, are confidential and proprietary information of Q/P Management Group, Inc. and can not be reproduced without the expressed written permission of Q/P Management Group, Inc.

Session Objectives

- Describe why organizations are using and measuring reuse
- Discuss how to measure reuse
  - Reuse of applications
  - Reuse of components
- Discuss how to account for reuse

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Why Utilize Reuse

- Improve developer productivity – write once / use often
- Reduce costs
- Improve quality
- Maximize system uptime
- Maximize testing resources
- Maximize developer capacity

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Why Measure Reuse

- Quantify the investment
- Estimate software development
  - Effort
  - Schedule
  - Staffing
- Measure the quality of reusable software
- Report the size of reusable software delivered
- Report “delivered” versus “developed” software
- Monitor vendor’s performance

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Types of Reusable Software

- Computer off the Shelf (COTS)
  - Examples: Oracle Financials/HR, SAP
- Common in-source systems
  - Example: A company builds a set of common applications to process customer transactions that are installed at multiple sites. All sites are required to utilize a core set of functions. Each site can be customized based on the unique business needs but no changes can be made to the core set of functions.
- Reusable plug-and-play components
  - Shopping cart functionality for an eBusiness store

Note: This is not intended to be an exhaustive list of reusable software

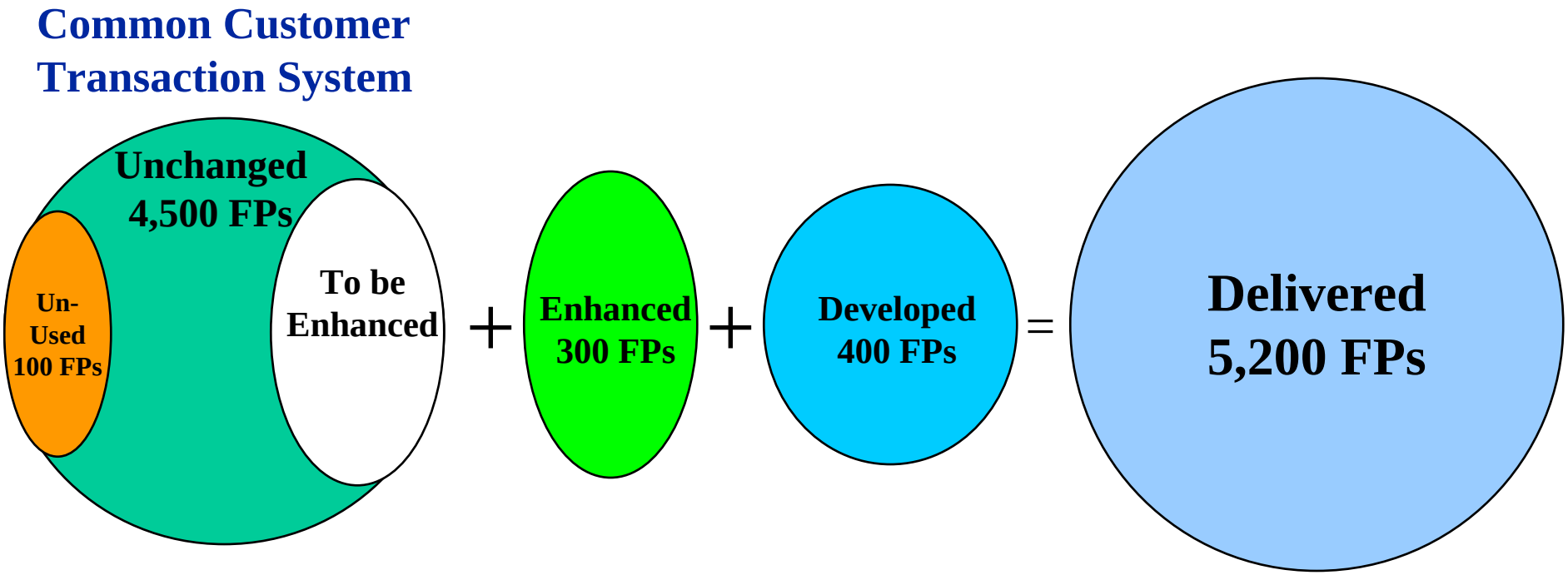
© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Measuring COTS and Common In-source Systems

- The approach is the same for both COTS and Common In-source systems
- Objective is to size all functionality
  - Size Out-of-the-box functionality utilized – not customized
  - Measure Out-of-the-box functionality requiring customization
  - Size additional developed functionality – customization for a site
  - Size Delivered functionality – Out-of-the-box + Enhanced + Developed functionality
- Measure productivity
  - Out-of-the-box functionality = FP size ÷ Hour to install/configure
  - Developed/enhanced functionality = FP size ÷ development hours
  - Delivered functionality = (Out-of-the box size + Developed size) ÷ (hours to install/configure + development hours)

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Example Measuring COTs and Common In-sourced systems



- Common Customer Transaction System Installation/configuration effort = 900 hours
- Developed/Enhancements effort = 14,000 hours

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Why Measure Reuse - Productivity

Reuse Project Productivity

- Out-of-the-box productivity
  - 4,500 function points ÷ 900 hours = 5 FPs per hour
- Developed/Enhancement productivity
  - 700 function points ÷ 14,000 hours = 0.05 FPs per hour
- Delivered functionality productivity
  - 5,200 function points ÷ 14,900 hours = 0.35 FPs per hour

Tracking each separately affords the opportunity to plan:

1) The overall effort required for future implementations
2) The effort required to install/configure the out-of-the-box functionality
3) The effort required to deliver the customized portion of the solution

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Why Measure Reuse - Costs

Cost per Function Point of Reuse

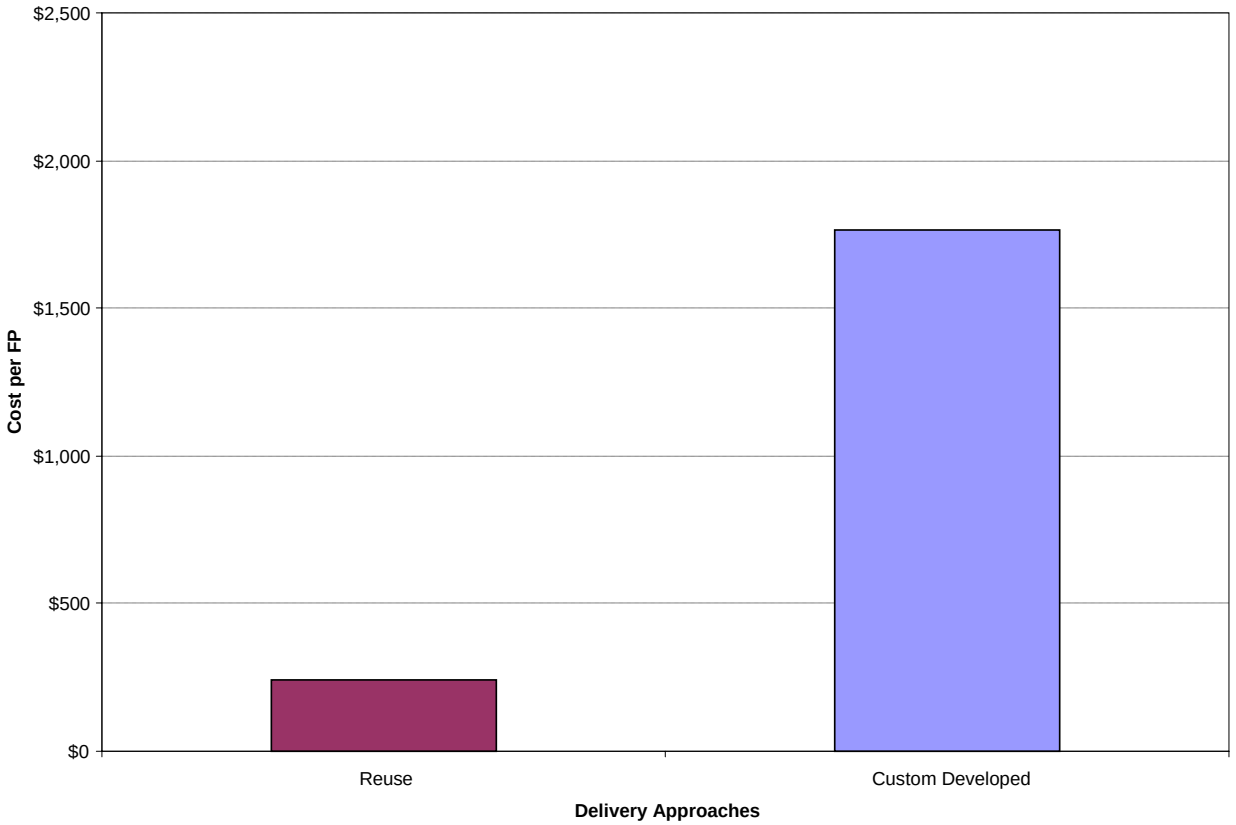
- Cost to Install/Configure Common Customer Transaction System
  - 900 hours * $65.00 per hour = $58,500
- Cost to Develop/Enhance the system
  - 14,000 hours * $65.00 per hour = $910,000
- Common Customer Transaction Acquisition Costs (spread across 20 installs) = $375,000
- Total solution costs
  - $58,500 + $910,000 + $375,000 = $1,343,500
- **Cost per function point**
  - $1,343,500 ÷ 5,200 = $258 per FP

Cost per Function Point Custom Solution

- Effort estimate of custom developed solution
  - 5,200 function points ÷ 0.037 FPs per hour = 140,540 hours
- Cost estimate for custom developed solution:
  - 140,540 hours * $65.00 per hour = $9,135,100
- **Cost per function point**
  - $9,135,100 ÷ 5,200 = $1,757 per FP

© Copyright 2007. Q/P Management Group, Inc. All Rights Reserved.

Why Measure Reuse – Cost Justification



Measuring reuse and comparing it to traditional development estimates helps to justify the financial investment

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Approach to Applying Function Point Counting Techniques to Software Components

Applying Function Point counts to plug-and-play components can be viewed from two perspectives

1. Function Point counts associated with the development of the component. Size the development/enhancement effort
2. The number of Function Points that a component contributes to the overall project/application that uses it

In both cases the Function Point size can be used to achieve the following for their specific environment

- Estimate software development
- Measure software quality
- Measure development/enhancement productivity
- Measure maintenance productivity

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Function Point Counting Guidelines for Components

- Recognize two user groups
  - Traditional end-users
  - Applications and/or components that interact with the component being analyzed
- Count all true end-user transactions that the component supports as if it were a traditional application
- Count all files that are being maintained within the component as ILFs, provided an elementary process exists for maintenance
- Any file that the component references that is outside the boundary of the application that the component resides in should be counted as an EIF
- If the component references another component's ILF that is within the boundary of the application that the component being counted resides in then the file should be counted as an EIF
- Count all transactions that exit the component being considered to communicate with other components contained in the application as either EIs, EOs or EQs

© Copyright 2007. Q/P Management Group, Inc. All Rights Reserved.

Function Point Counting Guidelines for Applications Using Component Technology

- Count only those transactions that enter and/or exit the application boundary as perceived by the end user of the application
- Count only the Internal Logical Files that are maintained by elementary processes of the application
- Count files that are maintained by other applications, but referenced by the application under consideration as EIFs
- Do not count files that were viewed as EIFs between components as EIFs. These should be counted as ILFs provided they are being maintained by an end user recognizable elementary process of the application. If they are not maintained then they should not be counted.
- Do not count files that existed within a component to store data for one or more elementary processes that were eventually sent to another component as an ILF or EIF if it is not recognizable to the end user

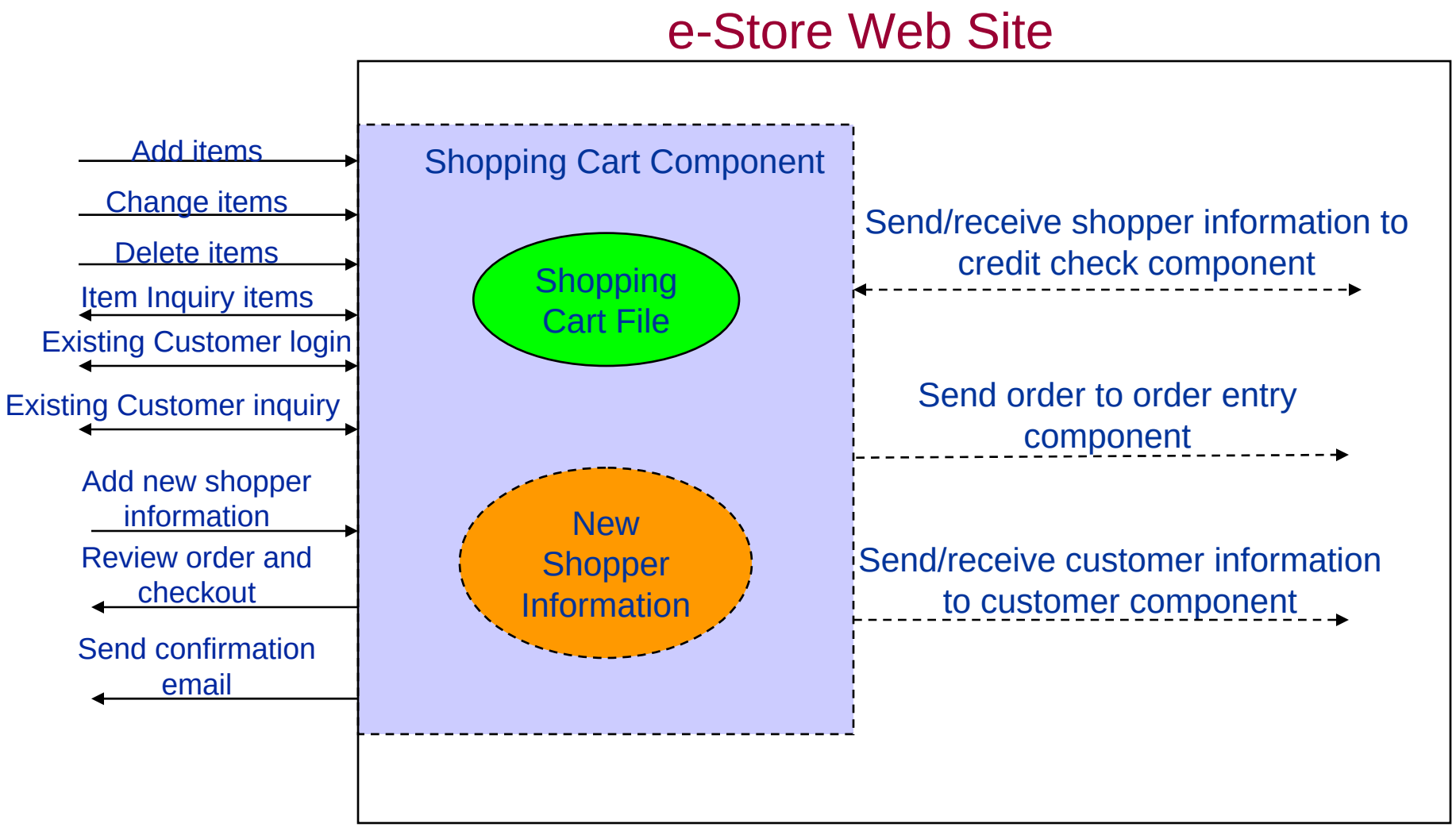
© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Function Point Counting Example #1 – *e-Store Shopping Cart Component*

- A software development organization specializing in e-Store web site development has developed a set of components that can be used to quickly construct web sites
- The Shopping Cart function will allow the shopper to add, change, delete, and view items selected for purchase. It will also allow the shopper to check out from the store and place the order.
- A developer will incorporate the Shopping Cart's functionality into e-store web sites to avoid the need to develop one from scratch
- The use of the Shopping Cart component will improve the productivity associated with creating the web site as well as reduce the effort associated with maintenance of web sites that use it

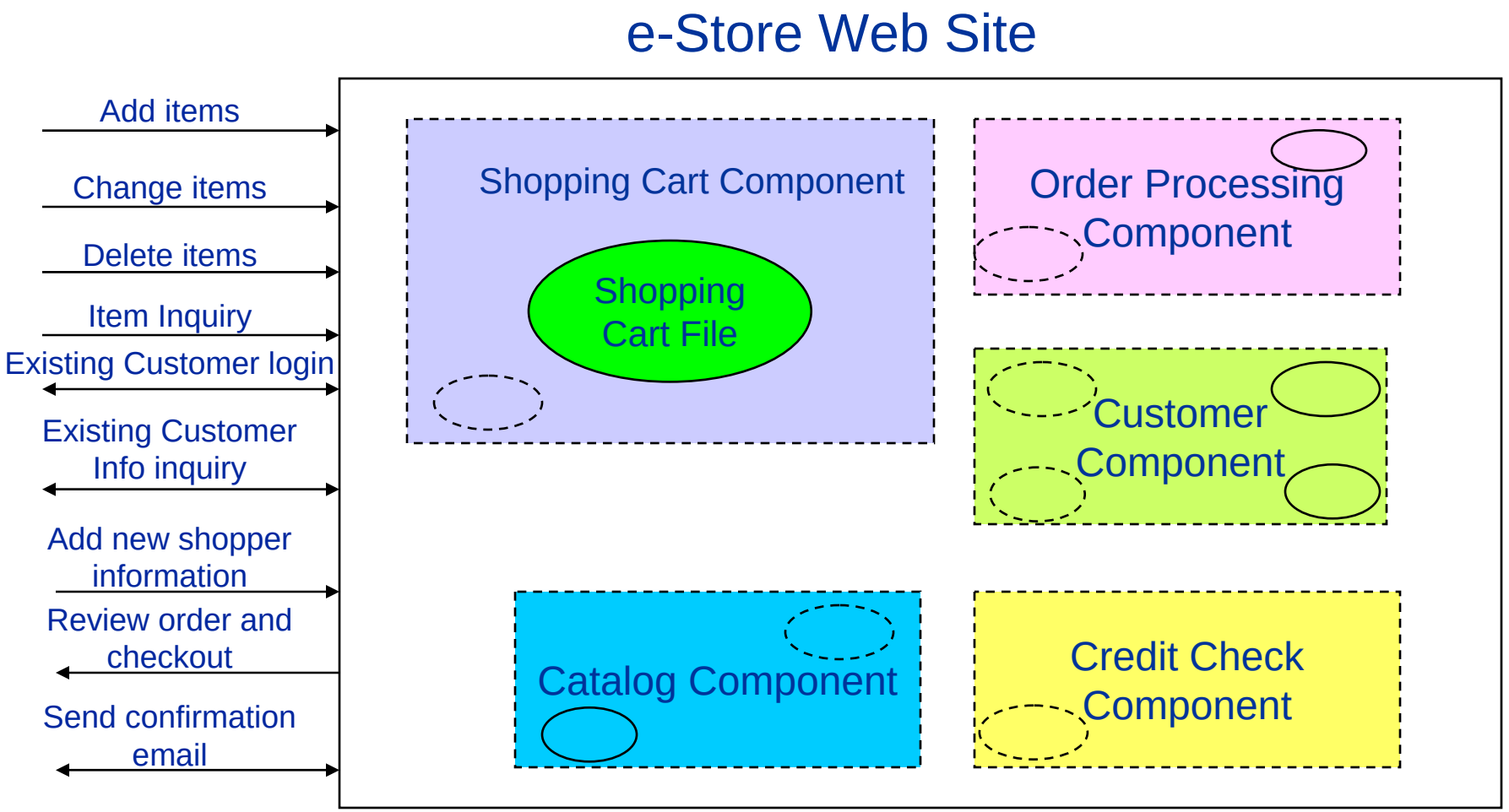
© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Shopping Cart Diagram – Component Developer



© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Shopping Cart Diagram End User Perspective



© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Project Size Comparison – Shopping Cart

Component FP Count	Type	FPs
Shopping Cart ILF	ILF	10
New Customer ILF	ILF	10
Add Item	EI	4
Change Item	EI	4
Delete Item	EI	4
Item Inquiry	EQ	4
Existing customer login	EQ	4
Existing customer information	EQ	4
Add new customer information	EI	4
Review order and checkout	EO	5
Confirmation email	EQ	4
Send/Receive Credit Check	EQ	4
Send/Receive Customer component	EO	5
Send/Receive Order Entry component	EO	5
Total	**71**	

Application FP Count	Type	FPs
Shopping Cart ILF	ILF	10
New Customer ILF	ILF	10
Add Item	EI	4
Change Item	EI	4
Delete Item	EI	4
Item Inquiry	EQ	4
Existing customer login	EQ	4
Existing customer information	EQ	4
Add new customer information	EI	4
Review order and checkout	EO	5
Confirmation email	EQ	4
Total	**57**	

Note: Assume average complexity

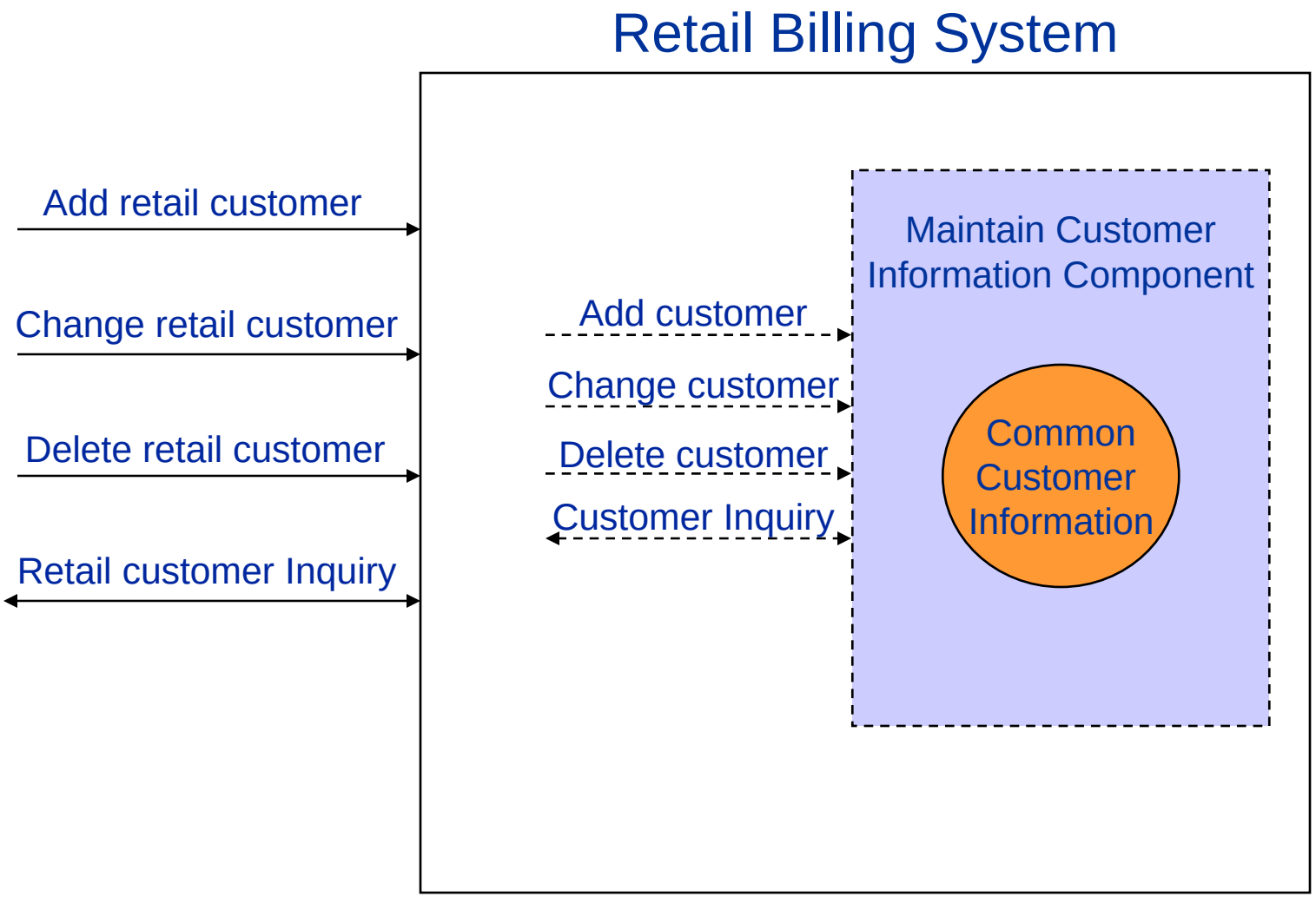
© Copyright 2007. Q/P Management Group, Inc. All Rights Reserved.

Function Point Counting Example #2 – *Maintain Customer Information Component*

- A regional utility company provides electrical and natural gas resources to both retail and wholesale markets
- The billing needs for these markets and services differ and have been built for the specific marketplace
- There is currently no way to evaluate the various relationships that a customer might have with the company
- The company has made a decision to move to a common customer relationship model which would bring all of the customer service information into a common data base
- Each business unit will remain responsible for maintaining specific customer information for their markets on the customer data base
- The direction is to build a common customer maintenance component that will be incorporated into each of the existing applications
- This module will allow the application to add, change, delete, and inquire on customer data specific to their line of business

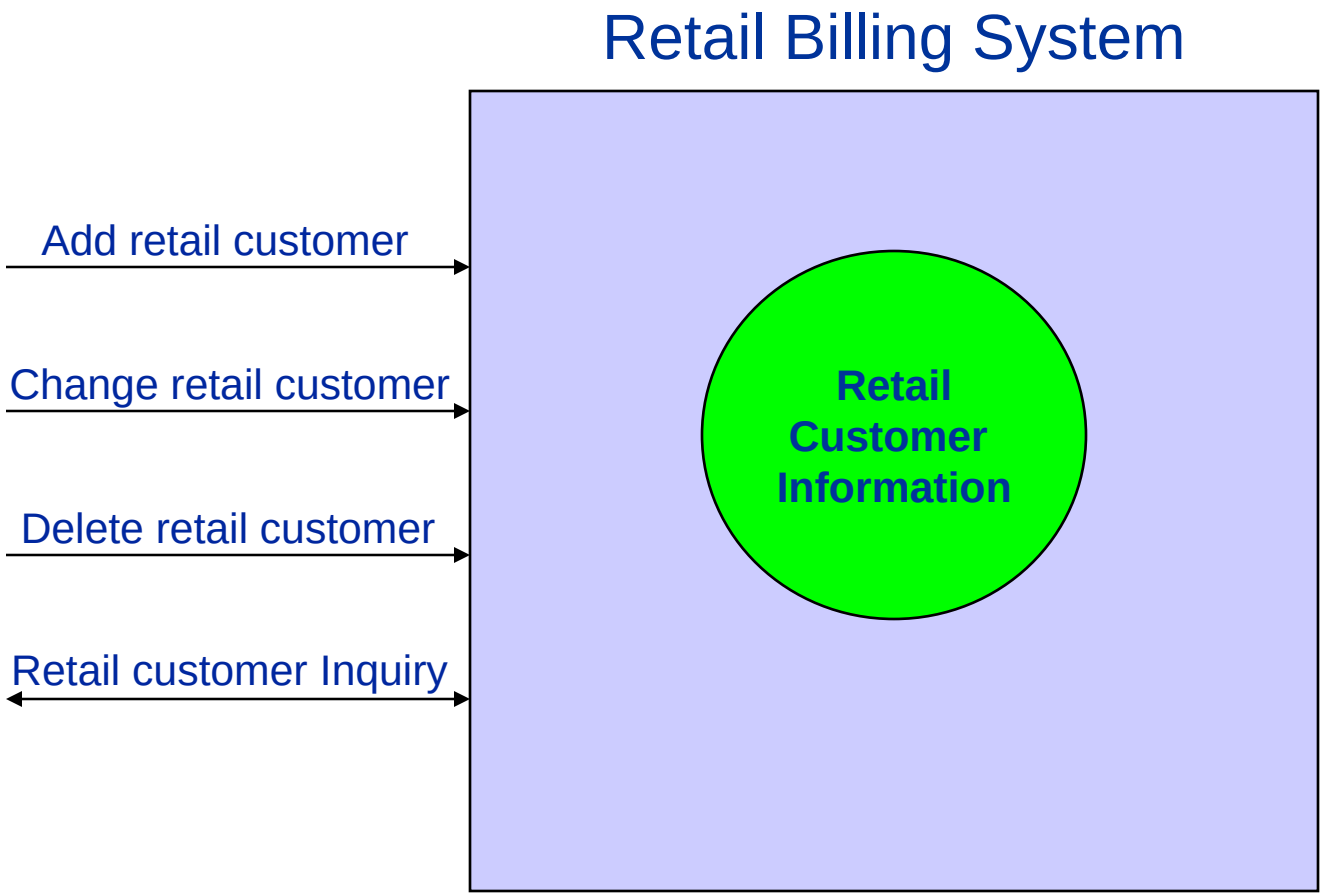
© Copyright 2007. Q/P Management Group, Inc. All Rights Reserved.

Maintain Customer Information Component Across Multiple Applications



© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Maintain Customer Information Component Across Multiple Applications



© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Project Size Comparison – Common Customer files

Function name	Type	FPs		Function Name	Type	FPs
Common customer file	ILF	10		Retail customer file	ILF	10
Add customer	EI	4		Add retail customer	EI	4
Change customer	EI	4		Change retail customer	EI	4
Delete customer	EI	4		Delete retail customer	EI	4
Customer inquiry	EQ	4		Retail customer inquiry	EQ	4
Total	**26**			**Total**	**26**	

Note: Assume average complexity

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Accounting for Component Software and its Contribution to the Business Portfolio

Component Portfolio			Business Portfolio	
Component Name	**FPs**		**Application Functionality**	**FPs**
Shopping Cart component	71		e-Store shopping cart functions	57
Common Customer component	26		Maintain Retail Customer	26
Total	**97**		**Total**	**83**

- It is critical to account for and measure the functionality within each portfolio separately
- Never add the portfolios together with the intent of representing a single view!

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.

Summary

Measuring reusable functionality offers many benefits to an organization

- Quantify the investment in reusable software
- Quantify the improvement in developer productivity and software quality
- Improve estimating accuracy

Why use function points to measure reuse

- Single measurement method for all types of software
  - No need to learn a new method or set of rules
  - A different view of the software
  - A different view of the end-user
- Flexibility of methodology allows for it
- Benchmark data available to support a measurement program

© Copyright 2007. Q/P Management Group, Inc.
All Rights Reserved.